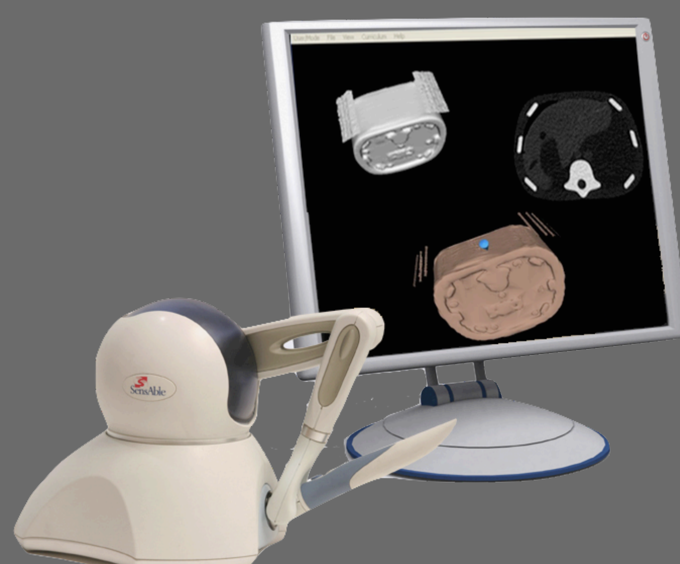
Real-Time Training Simulator for Medical Ultrasound Imaging

Team 7

Aldo Doronzo
Dag Atle Midttømme
Eivind Bjørkelund
Facundo Fortuny
Jelena Petrović
Simon Kiwanuka Takitse



NTNU
NO-7491 Trondheim

Preface

This report has been written during the course TDT4290 Customer Driven Project at the Norwegian University of Science and Technology (NTNU) during the Fall 2010. The class is part of the fourth year study at the Department of Computer and Information Science and is worth 15 credit points. The purpose of the course is to give students an authentic experience of a large IT project with a real customer. Students are divided into random groups and are assigned very diverse tasks, often on unfamiliar subjects.

The team truly believes that this project has given them the opportunity to both experience real problems but also to learn valuable solutions. Despite the challenging task, the team was able to accomplish it undoubtedly thanks to the support of both customers and supervisors. We would like to express our gratitude to SINTEF Medical Technology Frank Lindseth and Reidar Brekken, as well to our supervisors Meng Zhu and Tobias Buschmann Iversen.

Trondheim, 24th November 2010

Aldo Doronzo

Dag Atle Midttømme

Eivind Bjørkelund

Facundo Fortuny

Jelena Petrović

Simon Kiwanuka Takite

Summary

This report describes the work carried out by a group of students during the course TDT4290 Customer Driven Project at NTNU. This is a class where students are assigned a large IT project with a real customer. The project discussed is the development of a real-time training simulator for medical ultrasound imaging. The project sponsor was SINTEF Medical Technology, one of the most important research Norwegian player in this field. Finally, the report illustrates every phase of the project taking the reader from the project planning up to the final deployment and evaluation of the project.

Contents

Chapter 1

Introduction

This chapter introduces the project developed by some students attending the course TDT4290 Customer Driven Project at NTNU. This is a course where students are given the possibility to develop a large IT project sponsored by a real customer.

First a short description of the course TDT4290 Customer Driven Project will be given. In addition, the project sponsor Sintef Medical Technology will be introduced.

Secondly, an high overview of the project will be discussed. Objectives, stakeholders and time constraints will be presented.

Finally, the chapter ends by giving to the reader an overview of the report structure.

1.1 TDT4290 - Customer Driven Project

This is a mandatory course for all students attending the Computer Science program at NTNU. It is a 15-credit-points course that translates into an effort of 360 hours for each student.

The purpose of the course is to give students practical experience in the art of system development, and train them in how a real life work situation can be like. It involves all the phases of a real IT development process, such as: preliminary studies, project planning, requirements specification, system design, coding, evaluation and testing [29].

At the end of the course, groups will give a presentation of the system to customers, supervisors and to an external evaluator.

1.2 SINTEF Medical Technology

The customer and sponsor of this project is SINTEF Medical Technology. SINTEF is Scandinavia's largest research company that sells research based knowledge and services based in general technology, natural science, medicine and social science [1].

SINTEF's headquarters is in Trondheim, but they also have an office in Oslo with 350 employees, as well as minor offices in Bergen, Stavanger, Ålesund in Norway, and Houston (Texas), Skopje in Macedonia, Krakow and Warsaw in Poland and a laboratory in Hirtshals in Denmark.

SINTEF was started in 1950 by NTNU (then named NTH) as a prolonged reach towards the industry, and have continued to deliver technology to different areas of the industry such as oil ad petroleum, medicine and hospitals.

The company is organized into different areas of interest. The department involved in this project is SINTEF Teknologi og Samfunn which mainly focus on technology used in both medicine and health area.

1.3 Project Description

The project is named: "Development of a real-time training simulator for medical ultrasound imaging". The purpose of the project is to make a real-time simulator for ultrasound using CT-scan images as main source. At the moment, SINTEF has a similar simulator realized in Matlab, however, is both slow and does not support force-feedback haptic devices, which is one of the goal of the simulator that should be implemented in this project. In addition, the Matlab simulator gives a poor user experience mainly due to the fact that the user interface is made by simple sliders. This gives an unnatural feeling to the simulation.

An excellent simulator has been already deployed and is commercialized under the name ScanTrainer [2]. However, it is very expensive and the hardware used is tightly locked with the system. Our system should be more open oriented and should support several input devices on multiple platforms.

The main objective is to make a simulation program / framework that will support people in learning how to properly use a ultrasound probe. First, the simulator should be able to take as input medical images and render them into a 3D model. Then, using the special haptic device, the user should be able to "touch and feel" the 3D model and should be able to see the proper CT scan image and the corresponding ultrasound image. Using this simulator, students and medical personnel

will learn to discover diseases, internal bleeding and other issues. An effective training will help them to discover early complications that in turn may save the life of patients.

Since SINTEF has previous experience in this field, they will be able to consult our team with tips on which toolkits and languages are better to use to develop this kind of application. They suggest us to develop in C++ and to use both ITK and VTK toolkits since they are the de-facto standard in the medical imaging field.

1.3.1 Objectives

The main goals for the project are:

- Render a 3D model starting from a set of medical images;
- Extract the CT slice pointed by the user in the 3D model;
- Simulate an ultrasound image of the CT slice selected by the user;
- Develop a rather open and easily extendable framework;
- Deliver a detailed report containing all the information regarding the project.

1.3.2 Stakeholders

These are the stakeholders of the project:

Project Group

- Aldo Doronzo, aldodoronzo@yahoo.it
- Dag Atle Midttømme, dagatle@stud.ntnu.no
- Eivind Bjørkelund, bjorkelu@stud.ntnu.no
- Facund Fortuny, facundfortuny@gmail.com
- Jelena Petrović, jejapeta@gmail.com
- Simon Takite, kiwanuka@stud.ntnu.no

Customers

Frank Lindseth (frank.lindseth@sintef.no): Frank is senior research scientist at SINTEF Medical Technology. He has been working on ultrasound guided surgeries and navigation in neurosurgeries since 2002.

Reidar Brekken (reidar.brekken@sintef.no): Reider is a researcher at SINTEF Medical Technology. The previous version of Real-time training simulator was realized by him in Matlab. He has been working in this field since 2006.

Supervisors

Meng Zhu (zhumeng@idi.ntnu.no): Meng is a PhD candidate at IDI, NTNU. He is working in the Software Engineering group and is interested in mobile and social game, and Software Architecture.

Tobias Buschmann Iversen (tobiasbu@idi.ntnu.no): Tobias is a PhD candidate at IDI, NTNU. He is working in the Design and Use of Information Systems group.

1.3.3 Budget and Time Frame

The estimated workload is 24 hours per student per week. Since our group consists of 6 people the total workload will approximately be 1800 hours. The kick-off of the project is on the 31st of August and the project will continue until the final presentation on the 25th of November.

1.4 Report Structure

This report describes the work carried out during the project in the form of prestudies, requirements specification, implementation, evaluation and testing. Below, the reader can find a short description of the different chapters:

1 Introduction gives a short presentation about the course TDT4290 Customer Driven project, the project carried out by Team 7 and the report structure.

2 Preliminary Studies contains all the research and decisions regarding technical matters taken during the entire project.

3 Project Plan presents team organization, schedule of the project, and management rules used to conduct the project.

4 Requirement Engineering describes both functional and non-functional requirements of the application to be developed

5 Design Specification presents the system overview, an high level software architecture and associated constraints.

6 Quality Assurance contains the rules and routines used to assure a satisfying application quality.

7 8 9 Sprint describe each sprint, including goals, requirements, design, implementation, testing and evaluation.

10 User Documentation explains both the installation procedure and the main system functionalities.

11 Evaluation and Conclusion presents an evaluation of the product made by the team, the results and the process. Also, an evaluation of the course is provided.

Chapter 2

Preliminary Studies

This chapter presents all the research and decisions regarding technical matters taken during the entire project. First, it describes the considered development processes: Waterfall and Scrum. Secondly, it presents the knowledge studied to have a good understanding of the application domain. Thirdly, it discusses several technologies relevant to the project. Finally, in the last section, the reader can learn about the main technical decisions taken by the team to effectively develop the application.

2.1 Development Process

2.1.1 Waterfall

The waterfall model is a sequential software development process. It gets its name from the way it is executed, by following each phase sequentially downwards, like a real waterfall [51].

The phases usually are:

1. Requirements specification
2. Design
3. Implementation
4. Integration
5. Testing/Debugging

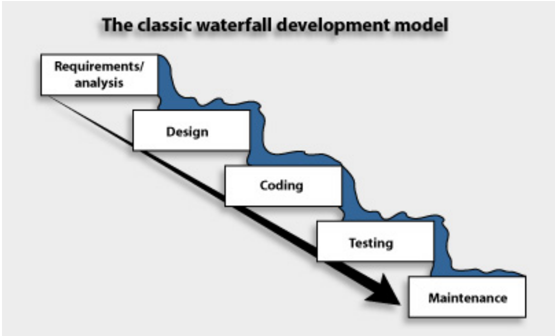


Figure 2.1.1: The classic waterfall development model. Source: Relativity-Corp.com

6. Installation
7. Maintenance

Each phase must be fully completed before you can move to the next, that means that the design has to be completed and finalized before moving on to the implementation phase. This is really demanding and requires a full and mutual understanding of all the requirements and how the system should work, and that all bugs are removed in the early stages, especially before implementation.

In the waterfall model, one usually implements the system in separate components, to remove some error risks, though this might create new error in later stages when components should be integrated with each other.

The testing and debugging phase is where the fully integrated system gets a thorough testing to see if every part of the system works with each other without bugs that might cripple the system. Installation is the process of installing the system on the machines where it should be used, this phase may also be just to hand the installation files over to the customer to be installed on which computer they prefer. This depends on the type of system.

Maintenance is not something the group will do with our system, but it is part of the waterfall model, where the team who worked with the system continues to improve the system, for example removing bugs that appear later, releasing new versions with new features and such.

2.1.2 Scrum

Scrum is an incremental, iterative method for product development, mainly used for large and complex systems and used with agile software development. The

scum method mainly consists of practices (sprints) and different roles [5].

The sprints are typically between two to four weeks long and consist of implementing small parts of the system that are called goals. These goals are high level requirements of work that needs to be done, and come from the product backlog. The product backlog is a list of prioritized requirements (you can see an example in Appendix E). Requirement priority can be low, medium, or high, and it depends from the importance of a requirement for the system and how important it is for the customer.

Deciding what will be in the product backlog is discussed between the group members on a sprint planning meeting. These goals will be based on the requirements the group find and those that the customer explains to the team. The group then needs to determine in which order and in which sprints the different goals are to be done, and the best scenario for one sprint is that the group complete all the proposed product backlog items for this sprint. During the sprint the product backlog is frozen, so no one can change the requirements. At the end of each sprint, any backlog products that was not completed will be transferred over to the next sprint and will be discussed on the next sprint planning meeting.

Roles

In scrum there are different roles based on which responsibility each member of the group has. These roles are:

Scrum Master: has the main task of being a buffer between the team and external elements. Mind that a Scrum Master is not a leader in that sense, but stands between the team and any distractions, for example by the customer or other stakeholders. Other tasks is making sure that the Scrum method is being used and that the team keeps focus on the most important tasks.

Team: the hard working members of the process, consisting of 5-10 persons. They do the design, development, testing etc. It is always a good thing that the team has different areas of technical experience so everybody in the team can give their opinions based on their expertise. This will ensure that the project will have a solid ground.

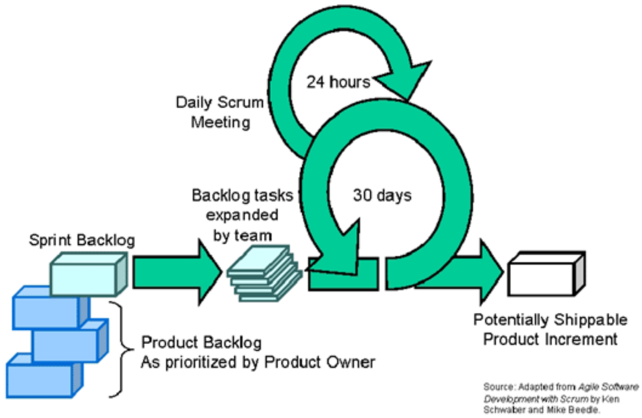


Figure 2.1.2: Scrum

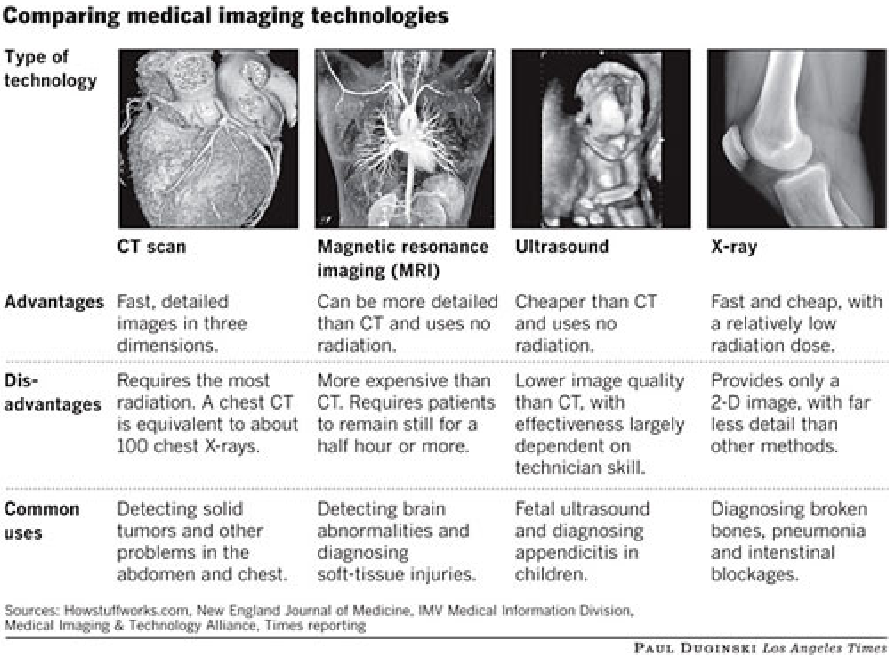
2.2 Domain Knowledge

2.2.1 Medical Imaging

Medical imaging is a technique used for the representation of the human body (or a part of the body) for clinical purposes and medical science [18]. This field covers radiology, endoscopy, thermography, microscopy and nuclear medicine amongst others. Medical imaging is one of the most useful diagnostic tools, since you can get a diagnose or a overview of the trauma without having to operate and open the patient up. All of these methods require very little of the patient, and are relatively safe. As this is a very large field in medicine, we will only talk about the different imaging techniques that is present in this project.

CT Scan

Computed Tomography (CT) is a medical imaging method [7]. It is far more detailed than for example ultrasound. It uses two X-ray beams crossing each other to produce a series of two-dimensional X-ray images of the body, which rotates around the body during the scanning. These images then will become the three-dimensional CT image of the body. They can be used to produce virtual images that show what a surgeon would see during an operation or a better image of the area of trauma before the surgery. That means that CT scans allow the doctor to examine the inside of the body without having to operate or perform unpleasant examinations. Detecting tumors and other problems are also achievable in CT scan images, and CT scans are very useful for scanning the head, where you can



Comparing medical imaging technologies

Type of technology	CT scan	Magnetic resonance imaging (MRI)	Ultrasound	X-ray
Advantages	Fast, detailed images in three dimensions.	Can be more detailed than CT and uses no radiation.	Cheaper than CT and uses no radiation.	Fast and cheap, with a relatively low radiation dose.
Dis-advantages	Requires the most radiation. A chest CT is equivalent to about 100 chest X-rays.	More expensive than CT. Requires patients to remain still for a half hour or more.	Lower image quality than CT, with effectiveness largely dependent on technician skill.	Provides only a 2-D image, with far less detail than other methods.
Common uses	Detecting solid tumors and other problems in the abdomen and chest.	Detecting brain abnormalities and diagnosing soft-tissue injuries.	Fetal ultrasound and diagnosing appendicitis in children.	Diagnosing broken bones, pneumonia and intenstinal blockages.

Sources: Howstuffworks.com, New England Journal of Medicine, IMV Medical Information Division, Medical Imaging & Technology Alliance, Times reporting

PAUL DUGINSKI *Los Angeles Times*

Figure 2.2.1: Comparing medical imaging technologies

see through the bone in the skull, which you can't with ultrasound. The downside to CT-scans are that the radiation is 100 times greater than with X-rays, and is very expensive to use [8].

Ultrasound Imaging

Medical ultrasonography is another medical imaging method [9]. It works by sending ultrasound (sound with a frequency higher than 20 000 hertz) towards the part of the body you wish to study. The sound waves are absorbed into the different body parts and are reflected back towards the probe that is emitting the waves. A computer then interprets these reflections to produce a image of the organs, bones and muscles inside the body, so in a way you can say it is a radar for medicine. This method of medical imaging relies on the different densities in the body, thus you can have a slice of the whole organ and not just the first surface the sound waves hits. Mind that bones are too structurally dense for the sound waves to penetrate, so there they will bounce of the bone, and on the screen you will just see blackness from the bone and downwards in the image. This means that ultrasound has a weakness when dealing with head trauma. Other than that, the use of ultrasound as a diagnostic tool is cheap, easy and not too hard to learn. It produces lower image quality than CT scan, but it is less expensive and does

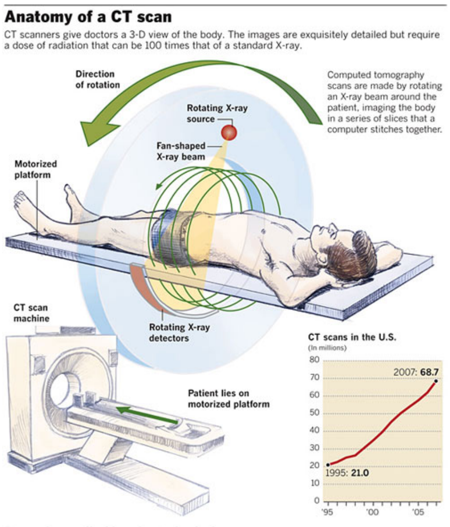


Figure 2.2.2: Anatomy of a CT scan. Source: Worldculturepictorial.com

not need any radiation in its use.

2.2.2 Haptics

Haptic technology incorporate the sense of touch and control into computer applications, by applying force, vibration and tactile feedback [10]. Using special input/output devices, called haptic devices, users can feel and manipulate virtual 3D objects. The type of feedback used to convey the sense of touch is determined by the type of haptic device being used [11]. Application areas for haptics are:

- Surgical simulation and medical training
- Painting, sculpting, and CAD
- Military applications
- Mobile applications (in the form of vibration response to touch)
- Gaming

2.3 Relevant Technology

In this project, the team should develop a real-time training simulator for medical ultrasound imaging, as it has been written before. This includes usage of Phantom

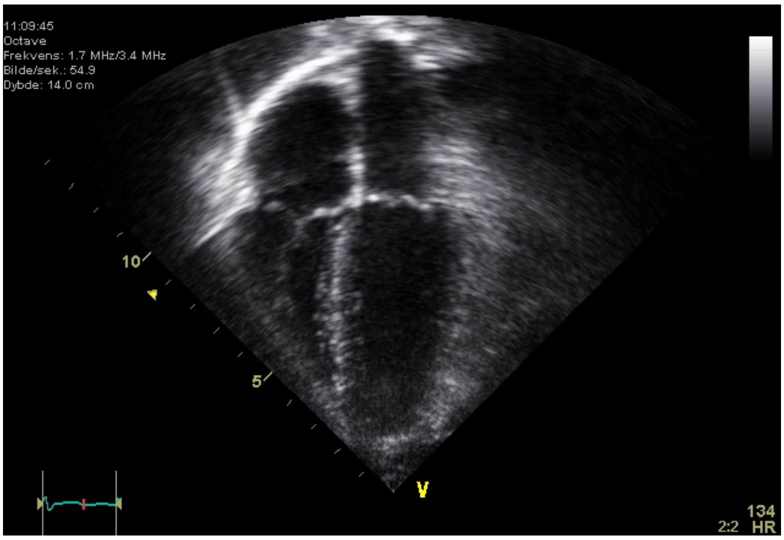


Figure 2.2.3: Example of an ultrasound image. Source: Wikimedia.org

Omni haptic device (see figure 2.3.1). The application has to be implement in a way that other devices also can be used later on.

In the first meeting, the customer introduced the existing system implemented in Matlab and suggested toolkits and technologies which the team should use during this project. All of these are described in the following sections.

2.3.1 C++

It is a language developed in the early 80s by the danish programmer Bjarne Stroustrup [12]. Developed with the basics in C, it was at first called "C with classes". But after years of development it got its more famous name C++ in 1983. It is regarded as a medium-level language, as it has features from both low-level and high-level languages [13]. It began by adding features such as classes, virtual functions, exception handling amongst some of the features and was standardized in 1998 as an official language of its own. It is now one of the most widely used programming languages to date.

2.3.2 OpenHaptics

The OpenHaptics toolkit includes the QuickHaptics micro API, Haptic Device API (HDAPI), Haptic Library API (HLAPI), Utilities, PHANTOM Device Drivers

Figure 2.3.1: Phantom Omni haptic device

(PDD), Source Code Examples, a Programmer's Guide and the API Reference [14].

QuickHaptics is a micro API that makes it fast and easy to write new haptic applications or to add haptics to existing applications. Built-in geometry parsers make it possible to set up haptics/graphics scene.

The HDAPI provides low-level access to the haptic device, enables haptics programmers to render forces directly, offers control over configuring the runtime behavior of the drivers, and provides convenient utility features and debugging aids.

The HLAPI provides high-level haptic rendering and is designed to be familiar to OpenGL API programmers. It allows significant reuse of existing OpenGL code and greatly simplifies synchronization of the haptics and graphics threads. The PHANTOM Device Drivers support all currently shipping PHANTOM devices.

2.3.3 Chai3D

Chai3D is a multi-platform, open source API for the Haptics devices, which also is written in C++ [51]. The main difference between this and the Open Haptics is that Chai3D is open source and it supports different Haptic devices, while the Open Haptics is made by Sensable and only for their Haptic device. It is a light platform which is easy to use and can easily be extended with other modules.

2.3.4 STL File Format

It is a file format for displaying images both 2D and 3D. This file format stores the image as a set of oriented triangles. It also includes a description showing which way the normal of each triangle is (which way the surface points), and then a description of the edges of each triangle. For large, detailed images, more description of each triangle is needed. STL-files can be stored in two different data formats, ASCII and binary.

ASCII has the advantage of being more descriptive in the file, using text to describe every aspect of the image, such as the header, number of triangles that make the image and then the information on each triangles.

Binary files store the same things, but in binary format. It includes an header of 80 bytes, a 4 byte information on the number of triangles and then the normal and 3 vertices for each triangle. Binary files are usually preferred, even though it is less descriptive than ASCII. But for large images, using binary data will make the file much smaller than the ASCII type, which is preferable since you rarely look in the file as a text so see the image [31].

2.3.5 OBJ File Format

Another file format for displaying both 2D and 3D images. This can also be viewed as a simple text file, which describes how the image looks. This format uses 3D geometry with x, y and z coordinates alone to describe where each vertice is to be, and all these vertices makes up the image. It is a simple language using "v" to start describing the vertice, "vt" for adding texture to the vertice and many more functions. This can also become a very large file for detailed images since the vertices needs to lie close and have different textures. You can also build faces which to lie texture on, which is a type of polygon. This is a simple file to build with some programs, but very difficult to understand which vertice that is where in the displayed image [32].

2.3.6 3DS File Format

Another file system for displaying 3D images, and uses binary data formats. 3DS uses chunks of data, each which includes an id for the type of data this chunks represents, a length for how long this chunk is and then the data for what part of the image and how that image is made follows. Parsers that don't recognize different chunks will skip them, which can make the reading of a 3DS file much faster. Each chunk makes a mesh of triangles with vertices, much like the other file

formats. The problems with this file format is that it has some limitations in the number of vertices and polygons (limited to 65536), and that textures are limited to the 8.3 DOS format [50].

2.3.7 CMake

CMake (short for Cross Platform Make) is an open-source, cross-platform system to build, test and package software, and is written in C++[15]. It is also used to make build files to incorporate different packages, toolkits and cross-platform builds into one project, as we did with incorporating the different toolkits into our project. ITK and VTK uses CMake to manage the configuration process and install the libraries in Visual Studio, which is needed to be able to use them.

2.3.8 VTK

VTK (Visualization Toolkit) is an open-source, cross-platform software system for 3D computer graphics, image processing, volume rendering and visualization[16]. It includes C++ class libraries and also several interpreted interface layers like Tcl/Tk, Perl, Python and Java. VTK supports many visualization algorithms and advanced modeling techniques. It also can be integrated with GUI toolkits such as Qt.

2.3.9 ITK

ITK (Insight Segmentation and Registration Toolkit) is an open-source, cross-platform development toolkit used for image analysis, written in C++[17]. It is mainly used for development of image segmentation and registration. Segmentation is the process of partitioning a digital image into multiple segments or sets of pixels. Typically the image is acquired from CT or MRI scanners, and in the medical sciences own image format DICOM. Registration is trying to transform different sets of data to one similar system. For example in medical industry a MRI scan may be aligned with a CT scan, so that we can combine the information in both.

2.3.10 DICOM

Digital Imaging and Communications in Medicine (DICOM) is a file format standard for handling, storing and transmitting information in medical imaging[18].

It also includes a network communications protocol which uses TCP/IP to communicate between systems. The DICOM format is widely used within hospitals worldwide.

2.3.11 Qt

Qt is a cross-platform application development framework used for the development of GUI programs. Qt uses standard C++ and additionally can also be used in several other programming languages via language bindings. Qt is free and open source software and is distributed under the terms of the GNU Lesser General Public License[20].

2.3.12 Integrated Development Environment (IDE)

The development environment is an important tool when developing application. It maximizes programmer productivity by providing an easy and user-friendly interface to help developers design, implement, compile and test the code.

Microsoft Visual Studio [19] is an integrated development environment (IDE). It can be used to develop application for platforms including Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, and Microsoft Silverlight. It supports programming languages such as C/C++, VB.NET and C#. It also support other programming languages like M, Python and Ruby through separate installation. Visual Studio is only available in Windows since it is a Microsoft application.

Eclipse is a free and open-source, multi-language software development environment with an extensible plug-in system. It is written primarily in Java and supports programming languages as Java. It also supports a lot of other programming languages through plug-ins like C, C++, COBOL, Python, Perl, PHP, Scala, Scheme and Ruby. It also supports other functionality like Latex and to create UML diagrams. Eclipse is released under the the terms of the Eclipse Public License which is an open-source software license [23].

NetBeans IDE [24] is a free and open-source integrated development environment written in Java. It supports Java, C, C++, PHP and JavaScript and other dynamic languages. It is also very modular like Eclipse, so it is easy to download new features such as support for other programming languages. It also is available on Linux, Windows and Mac OS X.

2.4 Evaluation and Conclusion

2.4.1 Development Process choice

We chose to use Scrum in our development process mainly for two reasons. First, for the flaws in the waterfall model, which is that the system needs to be perfect in each phase. Since our system consist of many different toolkits and subsystems that should be integrated with each other, and that for many of these we have very little knowledge of, it is hard to have a perfect design and implementation of each part of the system for each phase. A lot of the bugs will show during the integration and testing of the different kits and we will have to fix them as they arrive. With Scrum, we can more easily allow bugs to happen and fix them as they happens. This allows for more flexible programming and makes it easier for the group to fix errors that suddenly occurs.

The second reason is that there is very little flexibility in the waterfall model if the customer changes the requirements during the implementation phase. If a customer asks to add one more requirement or wants to change one of the requirements, the group almost has to start all over again because the design would have to change, and the new requirement may conflict with another part of the system. The scrum would easily handling this by adding the new or changed requirement to the product backlog for the next sprint. There would be the same work to do, but it will slide much more easily into the development process and then be faster done. Even with these bumps in the road, the customer still can get a finished and working product at the end of the development process.

2.4.2 IDE choice

All the IDEs support methods that increase efficiency like syntax highlighting, indentation functionality, easy browsing of existing code, list of functions in each class, auto-completion of code and different hotkeys. There are some minor differences between them, but this will not have any impact on the productivity of the team, since it will take very little time to adjust.

All IDEs are handy with debugging functionalities. This is very important so it will be much easier to find a bug in the system. They all support to run through the code step by step, to set breakpoints and evaluate parts of the code, realtime checking of syntax error and so on. Neither of this will have any impact on the decision, since they are so similar in this regard.

The customer made explicit demands that we develop the application in C++. Visual Studio comes with a native compiler, the two other does not. So if we

choose Eclipse or NetBeans IDE we will have to use extra time on installing a compiler. On the other hand, Visual Studio only supports Windows, and since there is people in the team that uses Linux and Mac OS, people will have to run Visual Studio in Windows as a virtual machine or install Windows as multi-boot. This will also take extra time.

We will use CMake to generate project files from ITK and VTK, so it is very important that the IDE supports that. NetBeans IDE does not have that support in CMake 2.8.2. There is probably a workaround but then we will have to spend more time figuring it out. CMake does, however, support Eclipse and Visual Studio. Also we need to be able to have good support for OpenHaptics and most of the examples and documentation for OpenHaptics uses Visual Studio.

In the end, we chose Visual Studio since it was very easy to set up, all the toolkits we were going to use supported it and had good documentation. Also the group members that used Mac OS or Linux, already had either dual-boot or virtual machine with Windows already installed, so the group did not have to spend any extra time on that.

2.4.3 Image file format choice

Choosing the right file format for this system was not only based on the pros and cons of the file formats themselves, but also on what kind of files the tools support. Both VTK and the Open Haptics (which is the tools currently in use for the image handling) supports 3DS, STL and OBJ files. However, the STL format was ruled out since it has some limitations with the VTK toolkit, in the way that you want to avoid reading the STL files as much as possible. STL files are big and slow to compile, so the team ruled out this file when this was discovered.

The teams main and difficult choice were between the 3DS and OBJ file formats. Both files are supported by the tools and have no large limitations which would tip the choice in one or the other way. It was finally decided to use the OBJ file format for the images. It was easier to use and convert the DICOM images to OBJ files within the system. It is also easier for the VTK toolkit to read and interpret the OBJ file format to render the 3D model of the medical images.

2.4.4 Haptic API choice

It was decided early with the customer that we should use the OpenHaptics API for the haptics device. This was because the OpenHaptics API is made from the

same company that created the Sensable haptic device which we use. In the start of the development it is also this API which were used in the first sprint.

Later one team member found another API which were possible to use, the Chai3D API. It supports more haptic devices, is open source and seemed more suitable to the requirements, were the system should support different devices. It also had a simulation of the haptic device with using an ordinary mouse as a replacement for the haptic device. This removed the need to have the real haptic device present when under the development process, which was a problem since there was only one computer in the group which could run the haptic device. This limited how fast the integration of the haptic device into the system would go. So there was some time where the Chai3D were used in development and the team tried to make it work with the system.

There was some serious problems that made the team to reconsider OpenHaptics though, which mainly was around the integration with the GUI and ITK. There is also the problem around that OpenHaptics have a smoother integration and a part of the API called "quickHaptics" which is a bunch of methods making development much easier to do and understand. Between the integration problems and the time used to understand Chai3D, the team decided to go back to and use OpenHaptics. It is only the customer who will be using the system and they have the Sensable device, so for the customer this won't be a problem. There was also the issue of time that made this decision, since the team couldn't spare any more time on these kind of preliminary studies and had to focus on making actual progress in development.

2.5 Related Work

2.5.1 IRIS 2000

IRIS is a cross-platform system to build a 3D model of a structure from any volumetric data set. IRIS allows users to load data taken from MRIs or similar images, and displays the data in different views (2.5.1). Three windows show the 3 orthogonal (2D) cross-sections and the fourth windows shows the 3D image.

It allows for the segmentation of data sets in these 2D slices and provides a 3D rendering of the data. The project goals specifically address the extension of the 3D display and manipulation capabilities of the system. It allows for the segmentation of a 3D data set, that is composed of 2D slices and further for the segmentation of data sets in these 2D slices to provides a 3D rendering of the data.

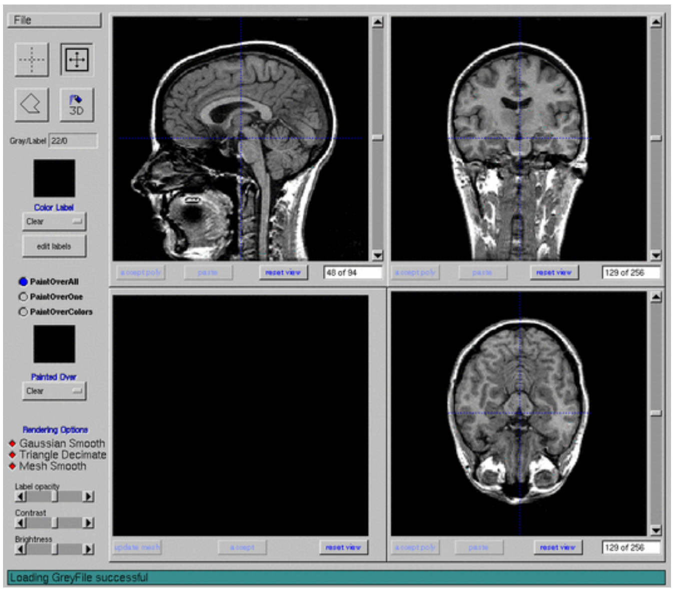


Figure 2.5.1: IRIS Display various image views of the data

2.5.2 MITK 3M3 Image Analysis

MITK 3M3 (2.5.2) is a versatile software application for medical image analysis developed by the German Cancer Research Center and built on the basis of a free open source software toolkit MITK which combines the Insight Toolkit (ITK) and the Visualization Toolkit (VTK). MITK 3M3 is independent of your system platform, whether you use Windows, Linux, or Mac OS X. As a toolkit, MITK offers those features that are relevant for the development of interactive medical imaging software some of which are lacking in ITK and VTK[43].

MITK 3M3 is endowed with features for medical imaging among which include DICOM support for importing from external discs and local databases. It visualizes image data in two dimension, three and fourth dimension [44]. This feature gives the possibility to view an image in different prospectives.

The medical imaging toolkit aims at providing support for an efficient application development of methods and applications dealing with medical images. Its scope comprises the broad range from the implementation and evaluation of custom research work, like for example algorithms, to the development of entire solutions for clinical studies or commercial use [46]. Its application is not limited to medical image processing tasks supporting medical diagnosis, but can also be used to support the development of therapy planning, and imaged-guided therapy applications[47].

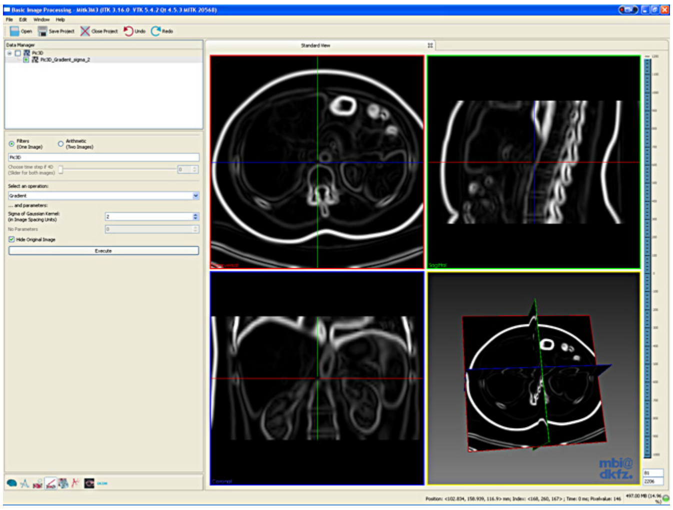


Figure 2.5.2: MITK 3M3 application for visualization of image data

Chapter 3

Project Plan

In this chapter, we will cover issues and decisions related to project management, organisation of the team, scheduling of the tasks, risks and requirement specification with use cases.

The first part describes team organisation. We have chosen a flat organisation of the team, where everybody have a equal authority, but every person have a different responsibility fitting their chosen roles in the group.

The next part covers the development model we chose, which is Scrum. The organization of the team and the scheduling of the development is fitted to the Scrum model. The overall schedule for the development, sprint plans and time division between the sprints are mentioned here as a part of the Scrum development model. The development plan will be the sprint backlog, as it contains all of the system parts we need to develop and implement and Scum don't emphasize on a lot of documentation and planning done before the actual sprint runs. The actual sprint and the product backlog may and will be adjusted along the development process, as is the custom in Scrum.

Further you will find a list and description of the most common and predictable risks this project has. A short summary of the risks, with the probability for each risk and how much they have on the project.

At the end of this part, the management rules are described. This part contains the templates and standards that the development process follows. This is the guideline for the team through the development of the system.

3.1 Team Organisation

The team have chosen a pretty flat structure because the team believes it will be more effective if everyone have some responsibility. This will make everyone feel that every team member have an important role to make a successful project and it also splits the workload more evenly on the team members. Figure 3.1.1 shows our organizational structure.

The team also chose Scrum as the development method for the reasons mentioned in section 2.4.1.

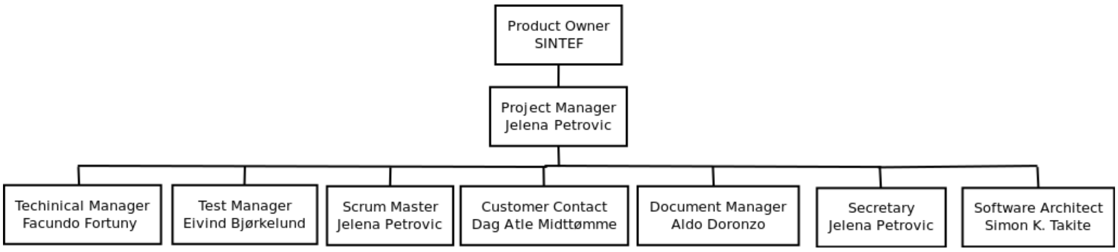


Figure 3.1.1: Organizational Chart

3.1.1 Roles and Responsibilities

Project Manager and Scrum master (Jelena Petrović) is responsible for the work progress and overall atmosphere among the team members. As a scrum master her obligation is to prepare and lead Scrum meetings and make sure that the usage of the Scrum framework is as effective as possible.

Technical Manager (Facundo Fortuny) is responsible for making decisions on the architecture and 3D engine, and to coordinate the use of different systems and tools to be used throughout the project. The system architect has an especially important role in the implementation phase.

Test Manager (Eivind Bjørkelund) is responsible for developing, executing and documenting the tests that will be used to check if the program works accordingly to the requirement specification.

Document Manager (Aldo Doronzo) is responsible for the quality assurance of the entire project documentation. Duties go from spell checking to layout definition.

Customer Contact (Dag Atle Midttømme) is responsible for the main contact with the customers, making sure their demands and requirements are taken

Table 3.1: Phases

Phases	Week Number	Planned number of hours
Project Planning	34-37	576
Sprint 1	38-39	288
Sprint 2	40-42	432
Sprint 3	43-45	432
Finalize report and presentation	46-47	288

into consideration and are up-to-date on the progress the group is making. He is also tasked with making appointments for progress status and general meetings with the customer.

Secretary (Jelena Petrović) is responsible for taking notes on the team, supervisor and customer meetings. Other duties are preparing weekly status report and agenda for supervisor meetings.

System Architect (Simon Kiwanuka Takite) is responsible for specifying the different architecture models suitable for the project and communicating high-level design decisions with the stakeholders.

3.2 Schedule

3.2.1 Project planning

This phase takes care of setting solid foundations for the entire project. After having prepared a good project plan, preliminary studies are going to be conducted to better understand the project domain. In addition, requirements are going to be gathered, analyzed and finally presented to the customer.

3.2.2 Sprints cycles

In this phase several cycles will be run in order to both develop the final product and produce the report.

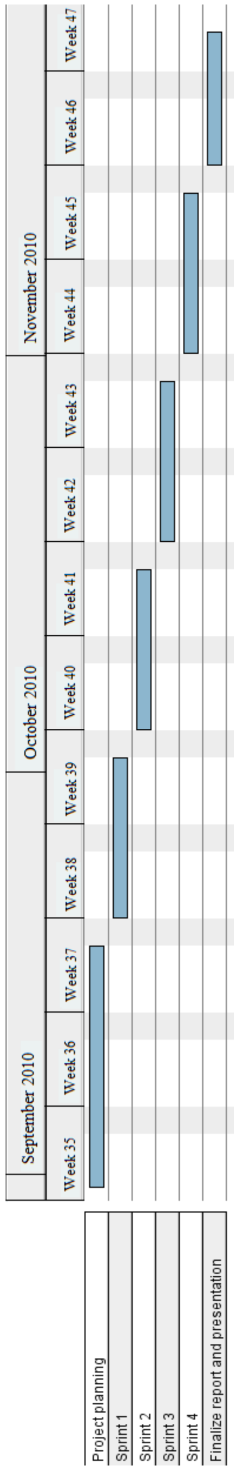


Figure 3.2.1: Schedule

3.2.3 Finalize report and presentation

The focus of this phase is to finish and deliver every part of the project. First, the customer should approve the final system. Afterwards, all the team's effort shall be spent on the project documentation and final presentation.

3.3 Milestones

Milestones is used to mark the critical points of the project work and help verify that the project is on schedule. Each one is described in the table bellow.

Milestone 1:	First Sprint Demo
Goal:	The goal of this first sprint is to make a basic GUI application based on the provided images we have from the customer. Basically just a framework of how the application will look. Also included in the application will be the rendering of medical images (DICOM) in 3D. The team should also be finished with the position information coming from the Phantom Omni device, and be able to try the device on the 3D rendering in the application.
Quality measures:	In this milestone the most important task is to integrate different technologies and toolkits, so we have a general GUI interface with a 3D object. It will only be an very early prototype which we can show to the customer.
Target date:	The sprint demo is held at October 14th.

Milestone 2:	Midterm Delivery of Report
Goal:	The report must include a copy of the abstract/introduction and pre-study and also show the general structure of the full report (Table of Contents).
Quality measures:	The report should not be too detailed, but must contain enough details to understand how each chapter is structured and what the final focus of each section should be.
Target date:	Pre-delivery of report should be delivered to the course coordinator on October 11th

Milestone 3:	Second Sprint Demo
Goal:	The goal of the second sprint is to render a CT slice image out of the 3D rendered image, and be able to show both the CT slice and the 3D model in the same screen. 3D rendering while changing the 3D object's position should also be smoother. The device support have to be extended in this sprint by having more universal support for different haptic devices.
Quality measures:	In this sprint it is important to render a CT slice image out of the 3D rendered image and extend the device support. It is very important to make the slice perfect and integration part can be passed to the next sprint.
Target date:	The sprint demo should be held at SINTEF on October 27th.

Milestone 4:	Third Sprint Demo
Goal:	The goal of the third sprint is to finish the product and deliver it to the customer. This means that integration between CT slice and Open Haptics libraries have to be done. Load and save a case feature shall also be implemented.
Quality measures:	The product has to fulfill the requirement specifications.
Target date:	The sprint demo should be held at SINTEF on November 22th.

Milestone 5:	Final Delivery of Report
Goal:	The report has to include all the necessary chapters and should not have more than 200 pages. The final report has to be finished a day before the target date, so it can be printed and ready before the deadline.
Quality measures:	The final report should have the layout and structure which follows the pre-delivery report structure. It should document briefly the work done during the project and should follow the standards for software documentation writing.
Target date:	The report should be delivered on November 25th.

Milestone 6:	Final presentation
Goal:	The final presentation should be prepared several days before the deadline. The project should be presented to the customer, the supervisors, the examiner and others that might be interested.
Quality measures:	Microsoft Power Point should be used for the presentation. The application has to run and work with the device as it is expected. All the members of the team have to be involved in the presentation.
Target date:	The project should be presented on November 25th at 1315 in IT 054.

3.4 Risk Management

The team have made a risk analysis of the project which shows the risks that can affect the project. The full risk table can be seen in Appendix C. The team have used a form of Probabilistic Risk Assessment (PRA), where every risk is divided in terms of the probability for it to occur and the damage it will do to the success of the project. A scale from 1 to 10 where 1 is low probability/damage and 10 is high probability/damage is used. After that the probability's score times the damage's score to get the risk is calculated. The risk will then be from a scale from 1 to 100, where 1 is low risk and 100 is great risk. This will give a good idea of what risk the team should really try to minimize or eliminate. Below is the highlights of the affluent risks, and the summary of all the risk as stated in appendix C.

Lack of Toolkit Knowledge (probability: 9, damage: 8)

Description Effective toolkit usage is essential to the success of this project because they contain complex functionalities the team is required to know in order to provide the correct program output. Unfortunately, the team has no previous knowledge about these toolkits. This could lead to greater risk of code either not working correctly or containing major bugs.

Solution To reduce the risk, everyone in the team has to read up on the different toolkits and learn the basic about them. In addition, all the programming activity will be conducted in smaller teams (usually of two people), so that if one programmer has a problem, the rest of the team can rely on the other one. If this latter will also not be able to find a specific solution, the entire team will have three times a week Scrum meetings where problems and possible solutions are going to be discussed.

Overloading Caused by Other Courses (probability: 7, damage: 7)

Description This project accounts for 15 credit points and in addition, all of the group members have 2 other courses which count for 7,5 credit points each. These courses do have a great variety of working amount, so there is a possibility that some of the other courses will take up a lot of time for some period. And this can lead to some delay in the project.

Solution The most important thing is to plan and organize well and divide the work load evenly on all the weeks. This will reduce the risk of some weeks having "too little" to do, and some weeks having "too much" to do. There also need to be oversight of when people have midterm exams and other assignments delivery in other courses, and that has to be taken into reconsideration in the project schedule.

Internal Misunderstandings (probability: 2, damage: 7)

Description The team composition is heterogeneous. All team member have different cultures, languages and backgrounds. This could lead to tremendous communication problems that might considerably slow down the project.

Solution The team is required to be really open minded. Practically, in order to achieve this goal, when something is not clear every team member must try to

express himself in multiple ways (oral, written, drawing, etc.). In addition, since the multicultural characteristic, the team must discuss problems really carefully and try to choose always the most appropriate pattern (direct, indirect).

3.5 Management Rules

3.5.1 Collaboration tools

Google groups is used for discussions and sharing the documents among the team members. All the documentation is written in one text document using Google docs. This draft is later after some reviewing, formatted in Latex. All team members are engaged in final report, and the Documentation manager is responsible for compiling the Latex document. All the charts and tables are, also made using Google docs and Use cases are created with a UML online tool.

3.5.2 Meetings

Regular Group Meetings

It was decided to have three regular group meetings weekly. The meetings are held on Mondays from 8:00 to 14:00, Wednesdays from 10:00 to 17:00 and on Fridays from 10:00 to 12:00. Those are not only informative meetings, but these hours are also working hours, because the team agreed that it is more efficient to work in the same room. The most of the team members prefer working in groups because they feel more motivated, a lot of misunderstandings are avoided in this way and group members can get to know each other better.

Supervisor Meetings

Meetings with supervisor are held once a week, on Wednesdays at 14:00. The duration of the meetings are usually one hour and all the documentation, including meeting agenda, weekly status report and meeting minutes from last meeting, had been sent to supervisor the day before. The secretary is in charge to take meeting minutes, prepare all the documentation and send this to the supervisor before Tuesdays at 15:00.

Customer Meetings

Meetings with customer are usually held once every two weeks, after each sprint. The project progress is demonstrated and guidelines for the next sprint are provided by customer, because the customer in this project has also a very important mentoring role, as well, since they have experience with 3D medical imaging. The meetings are held at SINTEF's offices. The customer contact is in charge to arrange the meeting date and time, and send an email as invitation.

3.5.3 Report Templates

Weekly Status Report

It is written by the secretary or the project manager on the end of each week. It contains activities planned for last week, activities accomplished during the last week, plans for the next week and a timesheet with working hours. The template of the document is included in Appendix B.1.

Supervisor Meeting Agenda

This is written by the secretary or the project manager before the supervisor meeting every week. It contains the place and time for the meeting, and also the topics which will be covered on the meeting. The template of the document is included in Appendix B.2.

Supervisor Meeting Minutes

This written by the secretary or the project manager after the supervisor meeting every week. It contains place and date and notes taken during the meeting. The template of the document is included in Appendix B.3.

Chapter 4

Requirement Engineering

The requirement specification gives a detailed description about all the requirements that have to be fulfilled based on the wishes of the customer. The chapter's structure is based on the IEEE 830 standard[21]. It includes an introduction, an overall description of the requirement and a more detailed, specific requirement section. It also has some use cases to show how the user will interact with the program.

4.1 Requirement Process

4.1.1 Introduction

This section will explain the process around finding, judging and estimating requirements for the simulator. This part is very important for actually making a good system, since the requirements define the core functions, both functional and non-functional, of our system. Having some clear and well defined requirements will help the team with splitting up the large system into smaller more easy to handle parts, and give some small goals during the development to accomplish.

The first section will explain how we planned our requirements and how we got them. This section will also explain how we judged the importance and workload of each requirement, as well as the estimation of how much time each requirement will take to finish and get working.

4.1.2 Planning the requirements

The process of getting the requirements was simplified by the customer. Right from the start the team got to know exactly what are to be made and the tools to make it with. The system is very specified in what should be done, it is a simulator of ultrasound by using CT slice images as a basis. The customer also provided with the tools that are to be used and some guidelines in how fast the simulator should work. Since we got to see an older version of the simulator made in matlab, we also got some ideas for requirements there.

After the first initial customer meetings, we started to break down the different general requirements from the customer into smaller requirements that would fit each sprint. This is done in team meetings with discussions, where the group as a whole figure out what in the requirements is most important and in what priority they should be done. The partitioning of the requirements into smaller parts were done in parallel with the preliminary studies of the toolkits the customer have told us to do, so the group would have a good understanding of how much time and work each requirement will require.

The planning usually starts with the basic requirement, for example starting with making a simple GUI for the customer to look at and so the team knows that this is the functionality the customer wants with the simulator. Then the team continue in later sprints with adding different functionality, increasing the complexity of the component. Like for the GUI we later start adding the other toolkits to the GUI and implements some functionality that the simulator should be able to do. The planning for later sprints is usually just to integrate everything into one product to deliver to the customer and make some final adjustments to complete the system.

4.1.3 Judging the requirements

Judging the importance of the requirements is done partially along with the customer. During the initial meetings, both internal and with the customer, the most important requirements were decided. Usually the most important requirements also are the requirements that take the most time, both in research and in actual doing, so those requirements got more people working on them than the smaller more easy requirements. Requirements like making the simple GUI is a small and somewhat easy task, and was completed in a couple of days. Mind that even if these requirements and tasks looks easy and takes little time, they are essential for the further development, since they also are an important part of the system. Other than this, the general importance of the requirements were done in discussions during the sprint meetings.

4.1.4 Estimating the requirements

Since the team was inexperienced with the tools, process and each other in the beginning of this project, the time needed for each requirement were difficult to estimate properly. The requirements that used VTK and ITK naturally got more time because of the need to learn about them as well, but requirements like the GUI needs less time to be estimated since it should go fast to make it. But exactly how much time that was needed for the requirements we could not quite know, so most of the estimations are done in a very general fashion, and we expect to see a lot of variations between estimated time and actually spent time on each requirement.

4.2 Requirements Introduction

4.2.1 Purpose

The purpose of the requirements specification is to get a clear overview about the requirements of this application. It is important to have a clear understanding of the requirements and to have a good and clean documentation of them. This is to make it easy for every one in the team to understand them, since this chapter will often be referred to during the implementation.

Obviously the user of the requirements specification will mainly be the customers at SINTEF Medical Technology and the team, but also the teaching supervisors and future users of the system will also be intended audience.

4.2.2 Scope

The system is intended to be a training simulator for medical images, so students and other people with limited experience in acquisition and interpretation of ultrasound images can train themselves with this simulator. More specifically the application should be able to read medical images and make a 3D volume out of them. The program should also be able to choose a medical image slice out of the 3D volume and be able to render an ultrasound image out of the image slice, all this while having support to navigate with a Phantom Omni device in 3 dimensions.

4.2.3 Overview

The rest of this chapter consist of an overall description of the product and the specific requirements. In the overall description chapter the product general factors are described. In the specific requirements section we list the functional and non-functional requirements and an overall use case diagram with some textual use cases.

4.3 Overall Description

This section describes the general factors that affect the product and its requirements, that includes product perspectives, user characteristics, high level requirements of the system and some constraints that will impact the project.

4.3.1 Product Perspective

The product is not part of a larger system, and is therefore independent and self-contained. The user of the system will use computers running Windows, Mac OS and Linux - with both regular mouse input and the Phantom Omni device. The application will require a screen resolution above 800x600 pixels and will have one user interface. The interface language will be English as the application will also be used by international users.

4.3.2 Product Functions

The application should be able to set up a 3D volume environment of medical images, and also to pick image slices and render some aspect of the image slice in ultrasound. The user chooses want he wants to do either with a mouse input or a Phantom Omni device.

4.3.3 User Characteristics

In this project there will mainly be only one type of user: the student. The student will be able to load the images, navigate the 3D model and render ultrasound simulation based on a image slice.

In the future there may be more users, like instructor to create different cases for the students to solve, and also a test/evaluation user mode where they can

evaluate the students. But the customer has told us to focus on the student user first, and if there is more time left, then start implementing new user modes.

4.3.4 Constraints

- The project has very limited time with a finite deadline at November 25th 2010.
- Limited experience with developing software among the team, and also no experience with the toolkits that has to be used. We are six members and not any more resources will be allocated to this project.

4.4 Specific Requirements

It is a complete description of the system behaviour. While functional requirements express the system features, the non-functional requirements impose constraints on the design and implementation such as: quality, performances and standards to be used.

Every requirement is assigned a certain priority:

- High: it is a fundamental need of the system and should carefully be implemented.
- Medium: it is an important need that requires to be implemented as soon as every high requirement is satisfied.
- Low: it is a "nice to have" requirement. If time and resources are sufficient, it will be implemented.

4.4.1 Functional requirements

ID:	Description	Priority
FR1	The user should be able to show a 3D image from the medical image (DICOM) parts in one window	High
FR2	The user should be able to load, save and switch a case	High
FR3	The user should be able to view the 3D image, CT slice and an ultrasound images in the same screen	High
FR4	Provide an interface for all the following input parameters: depth, width, gain, and frequency	High
FR5	The user can choose between several modes: student/training, instructor/setup and test/evaluation mode.	Medium
FR6	The user should be able to select from different curriculum options: Trauma, Fast, Ultrasound, Anatomy etc	Medium
FR7	The user can access help information from the help menu	Low

4.4.2 Non-functional requirements

ID:	Description	Priority
NR1	The application should run on Windows, Mac and Linux	High
NR2	The application should be able to render ultrasound images in real-time out of the CT slice image	High
NR3	The application should be able to render CT slice images in real-time	High
NR4	Make realistic ultrasound images from the CT slice image, where bones and different obstacles in the body is showed realistically on the screen	Medium
NR5	The application should support at least the following input devices: - Mouse (Provide sliders for input the parameters manually) - Phantom Omni Probe	High
NR6	The application should be able to give force feedback to the Phantom Omni Probe if it hits bones or other materials when the user navigates	Medium

4.4.3 Use cases

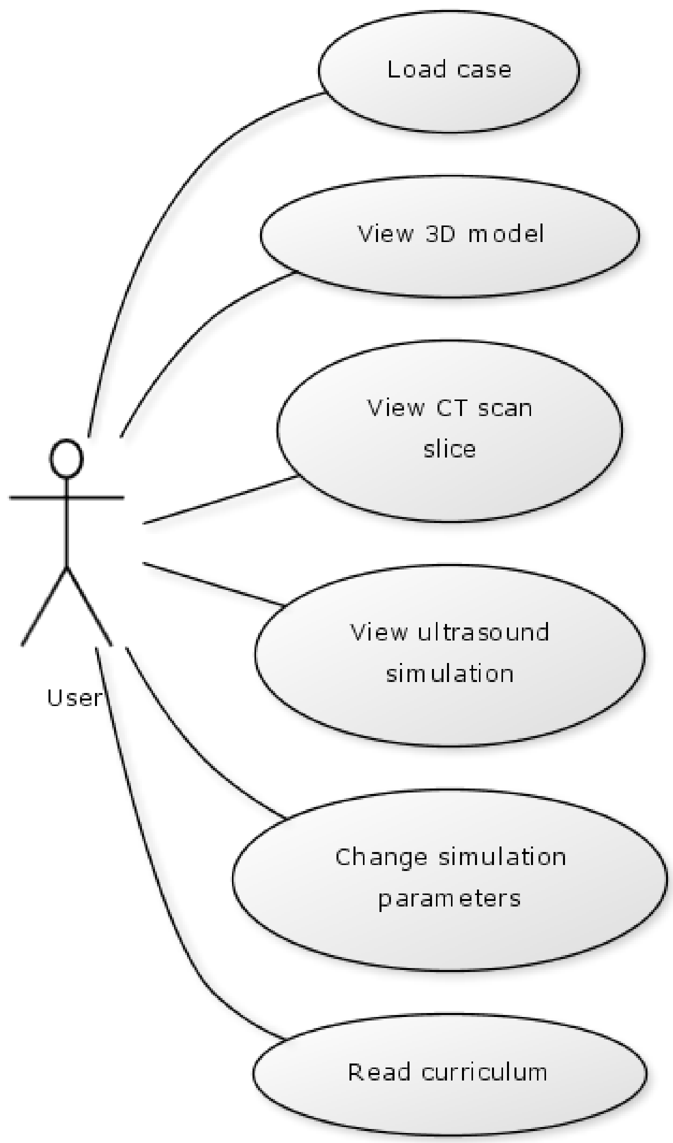
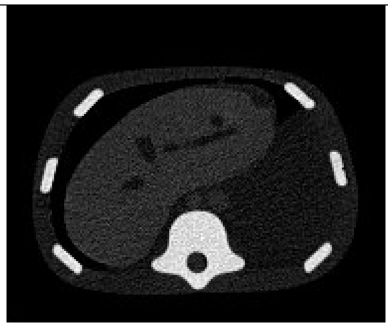
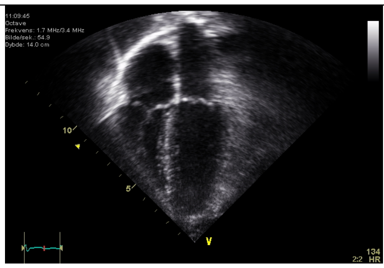


Figure 4.4.1: Use case diagram of the overall program

Name	C1: Load case
Actors	User, Phantom Omni device, system
Events	1. The user selects the menu item “File” and then “Load case” 2. The user selects viewing mode in “View” 3. The user moves the probe or the mouse to interact with the 3D object
Extensions	1a: If the case is not recognized, a message will be prompted .1: The user will be able to select another case

Name	C2: View 3D model
Example	
Actors	User, Phantom Omni device, system
Events	1. The user selects “View” 2. The user chooses “Show 3D”
Extensions	

Name	C3: View CT scan slice
Example	
Actors	User, Phantom Omni device, system
Events	1. The user selects “View” 2. The user chooses “Show CT slice”
Extensions	

Name	C4: View ultrasound simulation
Example	
Actors	User, Phantom Omni device, system
Events	1. The user selects "View" 2. The user chooses "Simulated US"
Extensions	

Name	C5: Change simulation parameters
Actors	User, Phantom Omni device, system
Events	1. The user selects an ultrasound simulation 2. The user can change the following parameters: depth, width, gain, and frequency
Extensions	

Name	C6: Read curriculum
Actors	User, system
Events	1. The user selects "Curriculum" in the menu 2. The user selects what kind of curriculum it wants information about 3. The application shows the information in a new window
Extensions	2a: If the information is not available, a message will be prompted

4.4.4 Requirement Change

There have been some changes in the requirements since the start of the development, mainly in the functional requirement part. During the second sprint, the

group and the customer discussed and agreed on changing and removing some requirements for the final product, since it became clear for both parties that the team could not finish all the requirements within the projects timescale. The most noticeable change is the removal of the requirement to view the ultrasound simulation. It was decided that this part would take too much time for the group to be able to do, as it would require much more time in finding the correct algorithms to use in the large VTK libraries, and then to integrate it with the system. So at this point we are delivering the rendered 3D model based on the medical images, from which one slice at a time is extracted and showed in the GUI. This is still a large part of the system they wanted though, as it still integrates the haptic device with the system. The requirement on changing the parameters has also been left out of the final system, as this deals with the ultrasound simulation that will not be in the system.

Chapter 5

Design Specification

Considering the preliminary studies and the requirements description in the previous chapters, the design specification shall demonstrate a practical description of the system upon which the implementation phases shall be based. The various sections and subsections shall follow the IEEE-1016 documentation standards for software design specification [24].

5.1 Introduction

This section contains an overview of the entire requirements specification, and also the general structure of the requirements.

5.1.1 Purpose

The requirement analysis in the previous chapter specified the issues the customer wanted to address in this project. The principle function of this system is to implement a real-time image simulator with the capabilities of displaying realistic ultrasound images and integrating the haptic device for interacting with the generated images on a desktop application. Users will vary in experience and tasks they have to execute. With efficient training, the users shall conduct various tasks such as trauma, ultrasound, anatomy among others.

5.1.2 Scope

In this document, we will describe the final design for the system. We define the functions and technologies to be used to achieve the solution, and also give a

detailed description of the components to implemented in preceding the chapters.

5.2 System Overview

5.2.1 Assumptions

For the purpose of non-invasive data acquisition, the use of computed tomography imaging and the routinely deployed injection of contrast agents were excluded. Therefore, it is assumed that the slices to be reconstructed into the 3D model are available before hand. This undermines the need for someone to take the computed tomography scans. During development, the team will not have their own image data, but it will acquire sample data-sets from the customer, which contains some slices data-sets generated by computed tomography, magnetic resonance imaging, and ultrasound. The elimination of computed tomography scanning meant that the need for professional medical skills needed to conduct such tasks unnecessary.

Generally, there are no professional medical skills required to use the application. However, to decipher more of what transpires within the application, it is important that the user have a clear understanding of the terms used in the application. Therefore, general medical knowledge is assumed.

It is assumed that the users for the system have got prior knowledge in tracked surgical tools, such as probes so they can interactively control the 3D model on the display. The system also requires that the user has minimal knowledge in executing simple tasks such as locating and loading images into the system or the use of other 3rd party applications such a Osirix [35]. Furthermore, it is assumed that the user can understand and perceive the images on the display as regards to the part of the body they were taken from.

5.2.2 General Constrainsts

The customer was keen on the system capability of supporting various navigation tools and other hardware. The problem is that most of the tools come with their own specifications on how to implement their functionality in various systems. At the time of development we are unable to implement and test all the navigation tools other than the haptic omni device. Although, we have provided for the extension of support for other devices in our architecture, the possibility of ensuring that these really work cannot be ascertained.

In the preliminary studies, we decided to use open source development tools in the implementation phase. Most of these tools do not offer support to the end user

and the tools of the APIs are in continual development hence they are prone to constant changes. The problem here is that the developers of the API change code and we have not provided for the possibility of accommodating these changes in our design specification.

There are important aspects of the system such as the accuracy of the image data to be fed into the application or the ease of use of the system. It is very hard to weigh these and thus impossible to be provided for in our architecture.

5.2.3 Platform

The system is devised on the principle that it shall be able to support and run on any platform. However, given the nature of three dimensional images, they demand a lot in terms of computation resources. If the computer where the system is deployed can support the display and running of 3D images, the system shall be able to run on most modern computer platforms such as Windows XP and above, Mac OS X and Linux.

5.2.4 Hardware

3D simulation requires a considerable amount of resources. Considering the available resources, we will not be able to test thoroughly how the system performs on different computer systems. We will develop and test mainly on laptops.

In-line with functional requirement NR5, the system shall support input devices such as mouse and haptic devices. We are using the Phantom Omni haptic device to make it possible for users to "touch and feel" (through force feedback) and manipulate the 3D model. The Phantom Omni model [39] is the haptic device the customer will provide. Its nature of portability and compactness make it possible for the group to move it to our working location. It uses IEEE-1394a FireWire port interface standard [38] that ensures quick installation and ease-of-use on our laptops.

Although the Phantom Omni device is IEEE 1394a complaint, there are reported cases where the device does not work with certain chipsets [36]. This is because of some manufactures implementation of the IEEE specification can vary widely. The system should work with any haptic device with the IEEE 1394a specification but it is recommended that they use those with the VIA chipsets [37].

5.2.5 System Environment

It is recommended that the application user possesses a generally good machine for running the application. This application revolves around the manipulation of three dimensional images which put a lot of strain on the computer resources. Medical imaging requires often a lot of physical memory in form of hard disk space [52]. For manipulation of the 3D image on the screen, a sensable haptic device is recommended but not required as the user will still be able to manipulate the model with a mouse.

5.2.6 System Architecture Components

The system will follow the layered architectural style and each component of the system is organized into the layers: the main application layer, visualization layer, image data processing layer and the hardware interface layer (see figure 5.2.1). The main interface layer will contain the graphical user interface that allows the users to interact with the system and will display the different views of the images from the visualization layer. The image data processing layer shall contain the processes taken to prepare the image for manipulation such as patient registration, segmentation, 3D reconstruction, handling of image data and position.

The processes here, just like in the visualization layer, are expandable. That is to say: the architecture allows for future development when more image processes are necessary. The bottom layer shall consist of the hardware interfaces that are used of reading the images into the system and for controlling the 3D image on the display such as the CT scanner, trackers among others. Also, here the architecture of the system allows for plugging in a number of hardware devices.

Visualization

All images data needs to be presented to the user in an appropriate way so to necessitate interaction. Based on the requirements and consequent customer meetings, the 3D volumetric data, shall be presented in various visualization methods such as surface, slice and direct volume rendering. Slice base rendering techniques present data in orthogonal slices in Axial, Sagittal and Coronal view of the patient data (see figure 5.2.2). The 2D Axial, Sagittal and Coronal views are displayed without additional processing from their original appearance, this is important for the user as it would provide more view options. The 3D model shall be rendered using a common technique where a series of 2D slices are saved into memory to generate the 3D image.

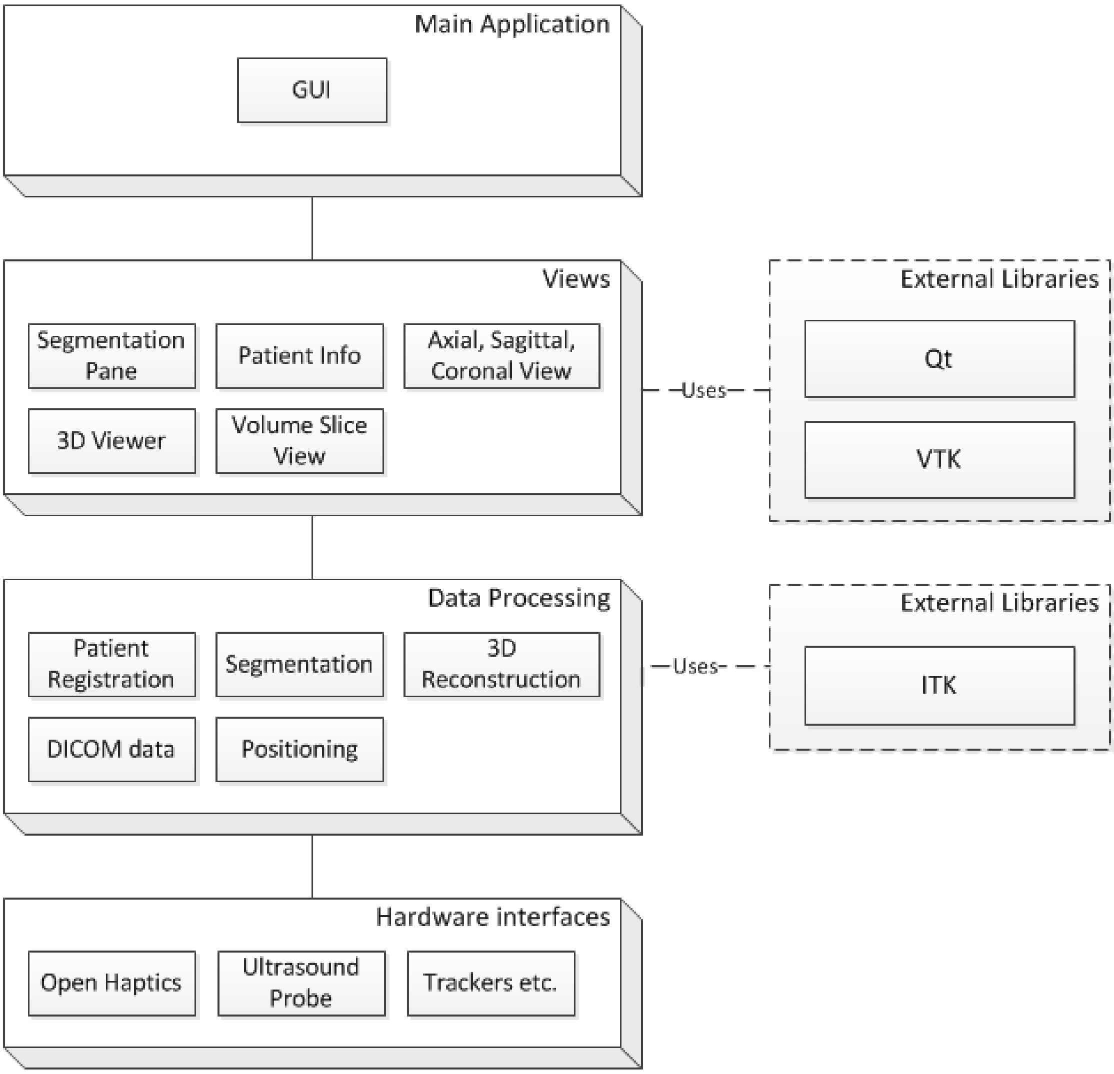


Figure 5.2.1: System Block overview

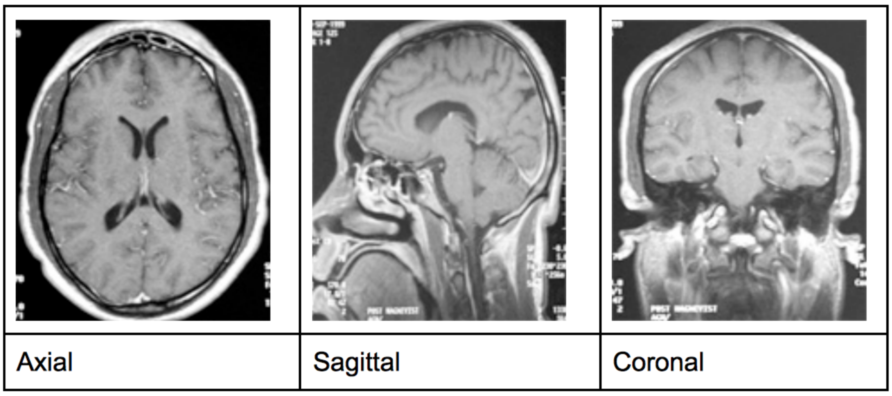


Figure 5.2.2: Axial Sagittal Coronal views

Segmentation

Input images have to be pre-processed before the segmentation phase. Once accurate data has been gathered, the process of extracting useful information from the images. Segmentation is a general concept meaning finding and extracting the features of the images cite [25]. A series of images in a DICOM format shall be read with VTK. A process of establish a common reference between the pre-processed images and the corresponding patient anatomy precedes from which a process for reconstruction a 3D model in VTK proceeds (figure 5.2.3).

The segmentation process shall use the VTK object framework for image processing. The framework provides a common interface for the physical location of the images and its geometrical properties. It further presents the relationship between the object that is independent of the form used to represent the object. From the use of the VTK framework algorithms, the solution shall allow the user to adjust the image to enhance the visualization and contrast of the anatomical part under study.

Registration

Registration is the process of transforming different data sets into one coordinate system. This will have to be done so that the real-time CT slice viewing correspond to the 3D volume coordinate system. Also the ultrasound simulation has to be aligned with the CT slice and 3D volume coordinate system.

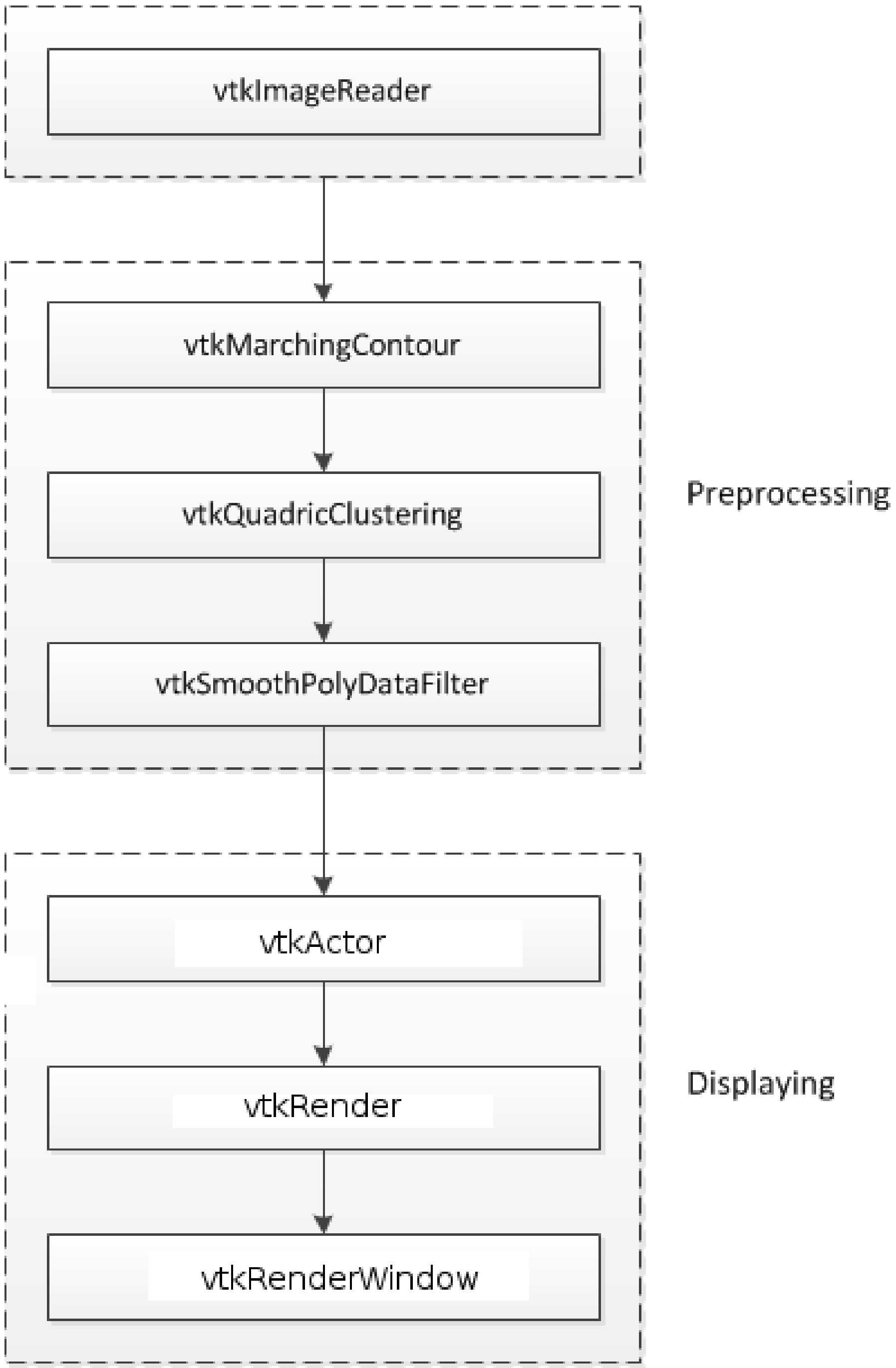


Figure 5.2.3: A pipeline for reconstructing a three dimensional image

Tracking

Optical positioning systems are some of the most accurate positioning systems with the most precise results [40]. The tracking systems are provided for under the hardware interface layer in our architecture. Although in the implementation, we will use the Phantom Omni haptic to intuitively interact with the 3D model, our architectural design allows for the extension of the system to accommodate all type of positioning systems for the measure of position and orientation with respect to a reference point or state such as acoustic, optical, mechanical and electromagnetic [41].

The phase for reconstructing the 3D ultrasound volumes and in navigated 2D is the probe calibration process. This is under the image data processing layer. This phase is met with one of the largest sources of errors for its reliance on probe calibration. This is the process of determining the position and orientation of the ultrasound image coordinate system relative to the coordinate system of the tracking sensor attached to the probe. The sensor attached to the probe provides the navigation system continuously with position measurements. An accurate calculation of this transformation is of crucial importance for: first, 3D reconstruction that preserves true anatomical shape and size in 3D ultrasound acquisitions, and secondly, navigation where the absolute position and orientation of the 2D real-time image is needed [42].

5.3 Architecture Rationale

The layered architecture style shall be used because it not only separates the user interface and the meta data, but also provides for the image data processing (logic) layer. The application layer provides a middle layer that allows the data files and the GUI components to be loosely coupled.

The application layer has to be modified if there are any changes to the format of the data files and the layer for the main application GUI will need little or no modification. This will make it easy for clients of this software to modify the data file format and attributes for further research purposes if they wish to do so. This layer makes the system more maintainable and reusable and also hides the complexity of processing data from the users.

The customer wishes for the system to accommodate different levels usages for example Student, Developer, Evaluation and Instructor and with a possibility of expanding this functionality to more user modes. The first user mode which the team will mostly focus on is the regular student user mode.

Considering the requirement NR5, the system should be able to support the mouse and the probe among other hardware components. The architectural consequence for this demand was to loosely couple hardware interfaces from the main application logic that is the image data processing. In doing so, this architecture will be able to accommodate a wide range of input devices with little or no changes to the other underlying layers.

Chapter 6

Quality Assurance

This section contains the rules and routines that the team followed during the development of this project. This is to make sure that the quality of the work, project and documentation is as the team expects it to be. This section describes these expectations and are solely the groups own expectations, although the supervisors and customers have been participant in the making of this. You will also find the test plan at the bottom of this chapter.

6.1 Routines

Routines are useful for the quality of the document and for the team because it ensures that the documents to the supervisors, the customers and the team is delivered on time and in the state both parts expects.

Group routines

- All weekly documents like the agenda, meeting minutes and status report shall follow the created templates. The secretary is responsible for this.
- All documents shall be written in English.
- Each individual team member is responsible for his or her part of the document and the document manager is responsible for the structure, merging and overseeing the final report.

Supervisor routines

- The agenda and all other weekly documents shall be sent to the supervisors the day before the supervisor meeting, that is within Tuesday at 15:00.
- Meeting minutes shall also be sent with the other weekly documents within Tuesdays at 15:00.

Customer routines

- Meetings will be scheduled by email at the end of each sprint by the customer contact, and must be sent out at least 48 hours before the meeting.

6.2 Test plan

Testing is essential to check if the final product will meet the specifications. Because of time limitation it was decided that each programmer are responsible for unit testing informally the code during the various sprints. Integration testing will be done by the coders in each sprint to see if the different parts work together. This will also be done informally. The system tests will be performed after each sprint, and the complete test plan can be seen in Appendix D.

The test manager is responsible for making the tests and the test plan, and will also have the authority to decide when the product has fulfilled the requirements so it can be delivered to the customer.

6.2.1 Overall Test Plan

Test	Importance	Phase	Description
System Test	High	Each sprint	Black box testing according to the goals or product backlog items chosen for this sprint.
Unit Test	Low	Each sprint	The unit testing will be done by each programmer to determine if the source code is fit for use. This kind of testing will be done on each functions, but will be done informally because of the limited time we have.
Integration Test	Medium	Each sprint	The integration testing will be done in each sprint to see if the different parts people have been working on will function together. This kind of testing will also be done informally to save time.

Chapter 7

Sprint 1

The first sprint started on Monday, September 20th where the team started the first sprint planning. On September, 23rd the first sprint meeting was held at SINTEF Medical Technology. The team and the customer agreed that the first sprint should focus on a basic GUI application with a 3D rendered model from the DICOM images. And also a possibility to use the Phantom Omni device to navigate the 3D model.

After the meeting, the group updated the sprint backlog and the system requirements so it was more in line with what the customers wanted. The full sprint backlog and system requirements can be seen in Appendix E.

The sprint started according to the plan. Although the team was unfamiliar with many of the tools, such as VTK, ITK, Qt and the general concept around medical imaging. Since much of the development process in the first sprint was organized for the team to implement while learning the tools, the team got a good understanding of them while working on the system. At the end of the first sprint the team had a simple concept GUI to show the customer, and managed to render the sample images using VTK and Qt.

7.1 Sprint Goal

The goal of this first sprint was to make a basic GUI application based on the provided images we got from the customer. It was basically just a framework of how the application will look. The rendering of medical images in 3D will also be included in the application. We should be finished with the position information coming from the Phantom Omni device, and be able to try the device on the 3D

rendering in the application. This should be demonstrated to the customer after the first sprint.

7.2 Sprint Backlog

ID	Priority	Description	Req.	Hours est.	Hours used	Responsible
P1	High	Basic GUI running, with basic GUI elements	None	7	7	Facundo & Simon
P2	High	View 3D model	FR1	84	80	Eivind, Facundo & Dag Atle
P3	High	Read the position information	NR5	7	4	Jelena & Aldo
P4	High	Integrate the Phantom Omni device with the application	NR5	66	66	Jelena & Aldo
P16	High	Integrate Qt with VTK/ITK	FR3	62	72	Facundo

7.3 Sprint Planning

The first sprint meeting was held at SINTEF Medical Technology on September, 23rd. The team presented the requirements specification and the product backlog. One non-functional requirement was added (NR6: Implement the force feedback to the Omni Phantom device) and one functional requirement was deleted from the specification (FR6: Delete a case). The product backlog was also updated according to changes in the requirements specification.

The first five top prioritised items are picked from product backlog and put in the first sprint backlog. The customers was satisfied and the demo will be presented in two weeks, at the end of the first sprint.

Basic GUI running Make a basic GUI application based on the provided images. It's just an example of how the application will look, but it will provide a basis for the implementation of the rest of the system.

View 3D image Visualizing the medical images in 3D requires good knowledge about the toolkits VTK and ITK. These toolkits have a lot of libraries and many different methods we can use for efficiently converting the 2D images into a 3D model. ITK or VTK will be used to load the images and to convert them together. VTK will be used to render the final result.

Read the position information In order to finish this task, the team should get familiar with Open Haptics toolkit. We should look into Quick Haptics API to check for examples with simple objects and how the position information is read. The next task is to try to read parameters of a 3D model in our own application using Open Haptics libraries. This part is described in the section "Integrate the Phantom Omni device with the application."

Integrate Qt with ITK/VTK To be able to show the rendered images in a graphical user interface, Qt and VTK/ITK needs to be integrated with each other. Since the installation of ITK, VTK and Qt with C++ took some time, the integration will probably take quite some time of the development process.

Integrate the Phantom Omni device with the application VTK and ITK should be integrated with Open Haptics libraries in order to use the device in the application. It means that research should be done on what formats are supported, and an appropriate export format for the rendered volume should be found.

7.4 Actual Process

Since the items in the product backlog are quite specific tasks in themselves, the group decided not to split them up. Mind that these tasks does not have many different elements in them, they just take a lot of time to do and demands some good insight in their proper use.

In the start of this sprint, a lot of time went towards installing, researching and understanding the tools. This was planned beforehand and therefore this sprint mostly went on schedule with the set of tasks. We were a bit suprised, though, by how long time it took to install all the toolkits. Installing everything took around 12 hours, but this were done mostly on the afternoon or the evenings, so it did not affect our planned schedule. Since both ITK and VTK are large libraries, it is expected that the research and testing of new methods and algorithms will continue throughout the whole phase of developing.

Basic GUI running Making a GUI is quite simple with the current versions of Qt. Qt has a Qt Designer which provides a simple and efficient way of making the GUI by drag-and-drop. The main process was therefore to understand how the customer wanted the system to look and to try to make the GUI intuitive and user-friendly. Since integrating Qt with the different tools is an own item in the product backlog, this item was quickly done early in the process.

View 3D image This task was the most time consuming during this sprint. In the start, many hours were spent on understanding the use of both ITK and VTK by going through tutorials and examples, since many of them are doing much of the same as this task. The classes and methods in ITK and VTK were closely examined to find the ones that are relevant to this project, so we did not "reinvent the wheel." This took a lot of time, more than original planned, since the libraries were bigger and much more advanced than first expected.

There are extremely many methods and algorithms to use, so much of the time in sprint 1 were spent on finding a good algorithm for 3D rendering, that is both fast and gives a good quality texture. Many of the algorithms are slow even though they produce a good rendering. But in the end we found an algorithm that were quite good, although it took some time to render the volume, but when it was first rendered, it performed smoothly.

Read the position information This task also involved some time used for just understanding and getting familiar with the toolkits, in this case . The haptics device is something no one in the team has tried before. Although some difficulties with having to travel to SINTEF to use and test the haptic device, the team made good progress in understanding the code and tools behind the haptic device and how the position information are being read with simple objects in QuickHaptics. We are know well on the way to be able to integrate this with the rest of the system.

Integrate QT with VTK/ITK After installing VTK/ITK and getting a basic understanding of those tools, the work on integrating VTK and ITK with Qt started. There were some problems with the integration because of different errors and then the process had to be restarted. The task was time consuming since the process itself took a lot of time. But it was completed in the right time and Qt are now integrated with the ITK and VTK tools.

Integrate the Phantom Omni device with the application There is a difference between what VTK, ITK and OpenHaptics supports of image types,

and it was necessary to find the appropriate file format to communicate between the haptics and the 3D model. After that, the haptics would be fully integrated with application. During the research, two types of formats were found to fulfill the needs of the application: STL and OBJ format. These two are the only ones that VTK supports, and they are also supported by the OpenHaptics libraries.

The next big problem the team had, was that the device could only be used by computers that have a FireWire 400 connector. On the team there were only two people that had a FireWire connector, and only one who had a FireWire 400 port. But the customer ordered an FireWire 800 to 400 adapter, so at least two computers were able to use the haptic device. Also we looked to other solutions, like searching for an application which can simulate the haptic device. Chai3D was able to do just that. According to the fact that Chai3D only supports OBJ and 3DS file formats, and OBJ and STL formats are supported by VTK, it was decided to use OBJ. The 3D image was exported to this file format and used in further development.

Due to unexpected problems with finding the appropriate export format, which took a lot of time, this task was not finished completely and was passed to the next sprint.

7.5 Implementation

The customer has set the guidelines for us to work along, so we did not spend much time of what tools to use, only on the actual implementation and integration of those toolkits. This was extremely time consuming. Just installing and compiling all the toolkits took around 12 hours. And often we got different errors and had to reinstall or recompile.

When everything was installed the team focused more on the integration by making the basic GUI as the skeleton of the system and with some form of 3D rendering in VTK in the Qt view. The actual system at this time was not even close to the real, final one. The team decided to develop two parallel applications having two different goals: one to manage the haptic device and one to render a 3D model from a series of DICOM images, all integrated with the GUI in Qt.

QuickHaptics Application This part was related with QuickHaptics libraries and communication with the haptic device. Figure 7.5.1 presents the UML class diagram. RenderingObjFile is our main QuickHaptics application class. The other classes are already included in the QuickHaptics libraries and are used in our

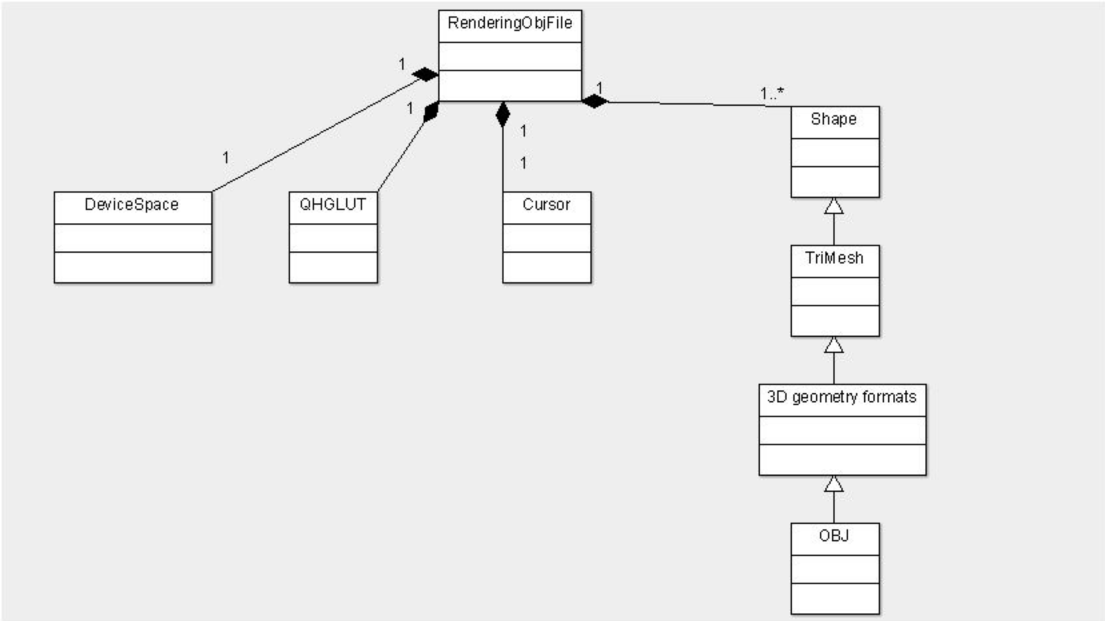


Figure 7.5.1: UML class diagram of the Quick Haptics application

project to show the model in OBJ file format and navigate it with the haptic device.

Main Application In this phase of the developing process, the goal was to render a 3D model starting from a series of DICOM images. The main application class is **GUI4**. The method **show3D** creates a 3D model of the imported medical images using VTK libraries. UML class diagram is shown in figure 7.5.2.

7.6 Testing

There was not any official time set for testing the system during this sprint, mainly because much of the system at this level is not a single program, but two different components by themselves that will make the system in the end. We were, though, able to integrate some 3D rendering with Qt, but it will need more improvement. With the haptic device, the team were able to read position information and start integrating it with our application, but we are not finished, so it was postponed to sprint 2. Some proper testing of that and other functionalities will be done in later sprints when all the different parts are integrated with each other.

Figure 7.5.2: UML class diagram of the Main Application

Test ID - Name	T1 - Basic GUI running
Description	GUI elements with a menu system shall be shown in a window using the Qt GUI framework. ITK and VTK libraries shall also be included.
Criteria	1. When the application is started, the user should be able to navigate in the menu system.
Estimated hours	69
Actual hours	79
Results	Passed
Comments	This was a very early prototype of the framework for the application, but the underlying design is in place which we can build on.

Test ID - Name	T2 - View 3D image
Description	The application shall be able to render the medical image parts in 3D.
Criteria	1. The user selects which medical images that will be 3D rendered. 2. The user shall then see the 3D rendering of the loaded medical image parts.
Estimated hours	84
Actual hours	80
Results	Passed
Comments	Managed to render and view a 3D image from the DICOM images, but needs optimization to get faster especially start rendering. Also the customer wants a surface rendering of only the skin, so it needs some improvements.

7.7 Results

At the end of the first sprint, 3D image view was ready. VTK/ITK libraries were integrated with Qt, and it was possible to see the model in the application, as shown in figure 7.7.1. Functionalities "CT slice view" and "Ultrasound image view" are not implemented yet, but test images were added, just to show the customer how the design of the application will look like when those views are implemented. The test images is the two images on the right side of the application window shown in figure 7.7.1. On the left is the 3D model.

The customer suggested that for 3D image rendering, surface rendering should be used instead of a 3D volume mapper. The reason is that it is not of the main interest what is inside the body, the surface or the skin is more important. Also, this change will make rendering quite faster since we can now eliminate everything inside the images. 3D image rendering will be improved in the next sprint.

The usage of the haptic device was demonstrated on an exported OBJ file (see figure 7.7.2), and this feature will be further extended in the application in the next sprint.

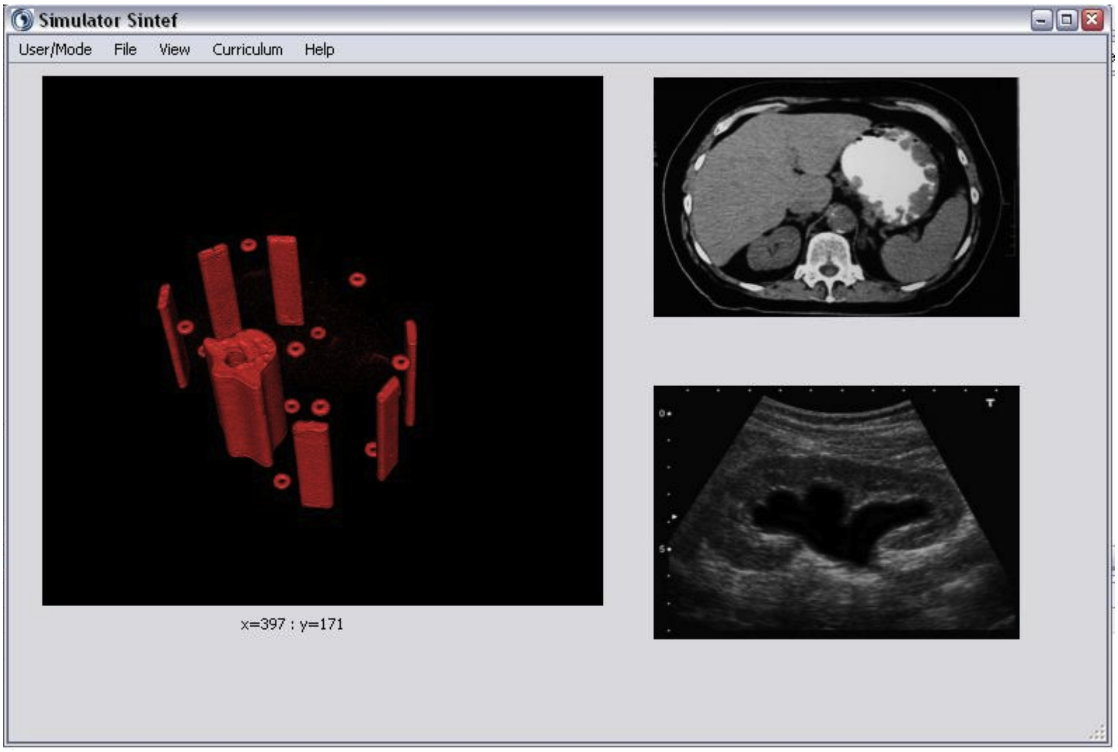


Figure 7.7.1: Screenshot of the Main Application (Sprint 1)

Figure 7.7.2: Screenshot of the QuickHaptics Application (Sprint 1)

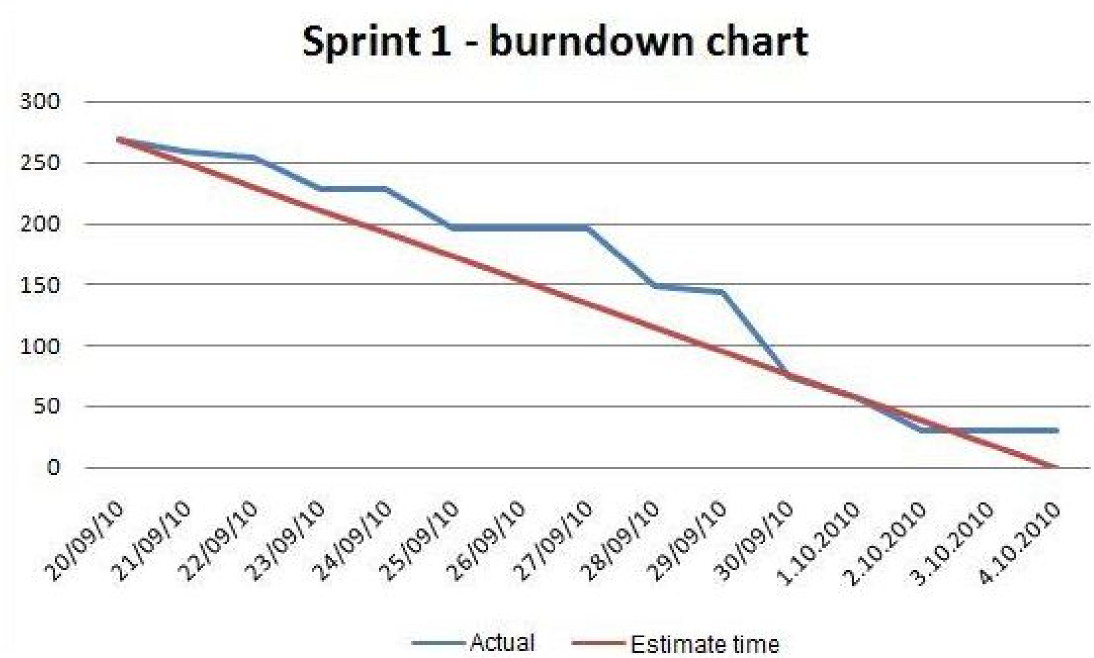


Figure 7.8.1: Burndown chart Sprint 1

7.8 Evaluation

According to the burndown chart, shown in figure 7.8.1, the team worked more than planned in the beginning. It was not possible to avoid because most of the toolkits were new and unknown to team members. We did try to estimate extra time for that, but we still managed to underestimate the effort it takes to learn all these new toolkits. After the first sprint we now have a basic understanding of the tools and we now know our skills better, so hopefully we will be able to estimate and delegate tasks easier in the coming sprints. We also have gotten more familiar with each other as a team, both socially and technically, so hopefully this leads us to work more efficent.

We achived most of our goals, but the sprint backlog item about integrating the application with the device has not been finished completely, and will therefore be moved to the second sprint.

Chapter 8

Sprint 2

The second sprint started on Monday, October 4th, and this chapter describes the work done during that sprint. The team continued with the work that was done in the first sprint. The first week was dedicated to write on the report, and make the necessary changes and additions to the report, since a pre-delivery of the report to external examiner and technical writing teacher shall be delivered on October 11th.

The customer meeting was held at SINTEF Medical Technology on October 14th. After the meeting the group updated the sprint backlog and the system requirements so it was more in line with what the customer wanted. Items P13: Render ultrasound images and P15: Realistic ultrasound images are moved from the product backlog. The reason was that the customer did not finish the development of the ultrasound simulator algorithm, so these items are excluded from the original product backlog. These features will be implemented by customer in the future.

The next four items are chosen from the product backlog and planned for this sprint: P14: View CT slice images, P17: Implement force feedback to the Phantom Omni Probe, P18: Optimize the 3D rendering and P19: Extend the device support. Also, one item from the first sprint has not finished yet (P4: Integrate the Phantom Omni device with the application). This one is also included in Sprint 2. The full sprint backlog and changed product backlog can be seen in the AppendixE.

8.1 Sprint Goal

The goal of the second sprint is to render a CT slice image out of the 3D rendered image, and be able to show both the CT slice and the 3D model in the same screen. In the first sprint it was noticed that when the user navigates the 3D rendering by

changing the position to the 3D object, it was a bit slow, so the team should also try to make that smoother. That will be done by using a surface rendering of only the skin instead of a 3D volume mapping of the whole object. After the first Sprint, the customer asked us to seek for several haptic environments to understand which one suited most their needs. Everything should be demonstrated to the customer after this sprint.

8.2 Sprint Backlog

ID	Priority	Description	Req.	Hours est.	Hours used	Responsible
P4	High	Integrate the Phantom Omni device with the application	NR5	30	30	Aldo & Jelena
P14	High	View CT slice images	NR3	72	72	Eivind
P18	Medium	Optimize the 3D rendering	None	64	52	Eivind & Facundo
P19	Medium	Extend the device support	NR5	64	70	Aldo
P17	Medium	Implement force feedback to the Phantom Omni Probe	NR6	68	8	Aldo

8.3 Sprint planning

The customer postponed the meeting scheduled on October 6th. Due to the new situation, the first week of the sprint the team has been working on the report. After the deadline for the pre-delivery report, the team continued to work on development of the application.

The second sprint meeting was held at SINTEF Medical Technology on October 14th. The next four items are chosen from the product backlog and planned for this sprint: P14: View CT slice images, P17: Implement force feedback to the Phantom Omni Probe, P18: Optimize the 3D rendering and P19: Extend the device support. Also the team will continue to work on P4: Integrate the Phantom Omni device with the application from Sprint 1.

View CT slice images The application should be able to view and render CT slice images in real-time. This process will involve finding usable and good algorithms for extracting and displaying the different CT image slices. The clue here is to find some way to extract the correct CT slice based on the position of the marker, either if it is a mouse or the haptic device. The group needs to find out how both VTK and the haptics use coordinates and how their coordinate system works and translate that to the CT slice coordinate system.

Optimize the 3D rendering After having successfully rendered the DICOM images to a 3D volume on display, it is now needed to improve and also change the rendering of the 3D volume based on feedback from the customer. The rendering should be in real-time and use surface rendering to render the outer part of the medical images, which is most often the skin. In other words that means the rendering of what is inside the object should be excluded. This shall make the rendering much faster when the user navigates the volume, either with the mouse or with the haptic device.

Extend the device support The current situation is that the tool we are using for the haptic device, OpenHaptics, only supports the products from the company that makes the toolkit, Sensable. A goal for this sprint will be to find other possible tools to include other devices that are on the market, for example try to use Chai3D, which is another open source haptic toolkit. This sprint will be used to find if this new haptic toolkit works and if it can be integrated with the system.

Implement force feedback to the Phantom Omni Probe This task is about exploring the solutions which give the user feedback while navigating a 3D model with the haptic device. In order to achieve this goal, OpenHaptics libraries and also Chai3D should be examined in this area.

Integrate the Phantom Omni device with the application (continued) Most of the research work was done in the first sprint. What is left is to integrate exported OBJ files with the application and test it with the device.

8.4 Actual process

In this sprint, the team started working more closely together. Each member have used the first sprint to study their part more closely, so in this sprint the members

will share the knowledge within the group and start integrate the two separate parts with each other.

View CT slice images This turned out to be a huge challenge. Early in the sprint the team managed to view the CT images and go through the slices using the mouse coordinates in the window where the images were shown. The problem is that the algorithm is far from perfect, for example for the time being the system are not able to show all the slices, only the top ones. That is something we have to look more into in the next sprint since we ran out of time in this sprint. The plan further will be to extract the correct CT slice from the medical images based on the coordinates of the haptic device on the 3D model, making the right coordinate correlation between the two. This part (View 3D image and CT slice) is left for the next sprint.

Optimize the 3D rendering The team managed to make the image rendering go much smoother during this sprint. A joint effort by choosing a better algorithm for rendering by only extract the surface and render that. Also the team was able to export the DICOM images to OBJ file format and using the OpenHaptics toolkit on our created 3D model. Further optimization of the images will not be made since for the next sprint, there is much more pressing issues on the table.

Extend the device support Different toolkits other than OpenHaptics have been tried at this point. It seemed Chai3D was an alternative to OpenHaptics that supported both SensAble and non-SensAble haptic devices. However, it was both slower and worse in rendering. Chai3D uses a more general haptic algorithm than OpenHaptics, which specializes only on SensAble haptic devices. Chai3D (figure 8.4.1) also produced a 3D model in a lower quality than OpenHaptics (figure 8.4.2), which is truly an important disadvantage. Since the customer had only SensAble haptic devices, the team decided to not spend more time on any other haptic toolkits than OpenHaptics.

Implement force feedback to the Phantom Omni Probe After some research, the team realized that force feedback is implemented in the QuickHaptics. So the only task to be done is to integrate the Phantom Omni device with the application and the QuickHaptics toolkit takes care of the rest.

Integrate the Phantom Omni device with the application (continued) In sprint 1 we had chosen OBJ file format as the appropriate export file type, so

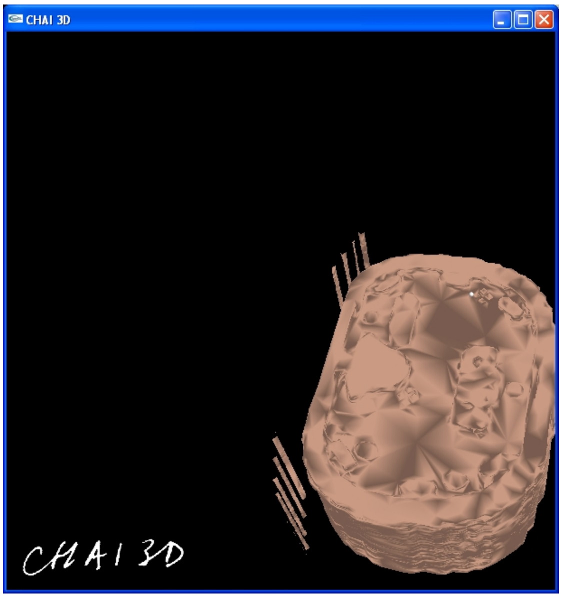


Figure 8.4.1: Navigating the object file in Chai3D

Figure 8.4.2: Navigating the object file with Quick Haptics

in this sprint the team were able to load 3D models and connect the device to the rest of the system using QuickHaptics. QuickHaptics is a part of the OpenHaptics toolkit, and is what the team used to integrate the Phantom Omni device with the application. It provides simple methods like being able to navigate and transform the 3D-model and it includes force feedback when touching the 3D model.

There was one problem, however, with the integration. Since it was decided to use OpenHaptics libraries for implementing the haptic device, there were a problem integrating the QuickHaptics with the Qt GUI. The team tried to find a solution, but it was stated several places online that it was not supported in QuickHaptics. We even asked on the SensAble forum if it was possible and they simply answered that it was not supported in QuickHaptics. So, in order to get full integration with the Qt GUI it was needed to use the full API of OpenHaptics and write it ourselves. This alone is a 3 month project so it was decided to launch the haptic navigation in a separate window which just communicates with the main Qt GUI. The CT slice images will still be shown in the main window.

8.5 Implementation

When the sprint was finished, the team was mostly on time with all the set goals and the system is starting to come together at this point. In figure 8.5.1 the class diagram of the current application is shown. In comparison with the previous sprint, system is integrated in one application now. Classes CTSlice and OpenHaptics are added. The first one deals with showing CT slices of the given 3D model. OpenHaptics class is integrated from the previous sprint where it was a separate application. Also, rendering part is now moved in the separate class Render3D.

8.6 Testing

The testing of the integration of the haptic device with the force feedback functionality is done in this sprint since that has now been integrated in the application. Also we tested the new optimized 3D rendering.

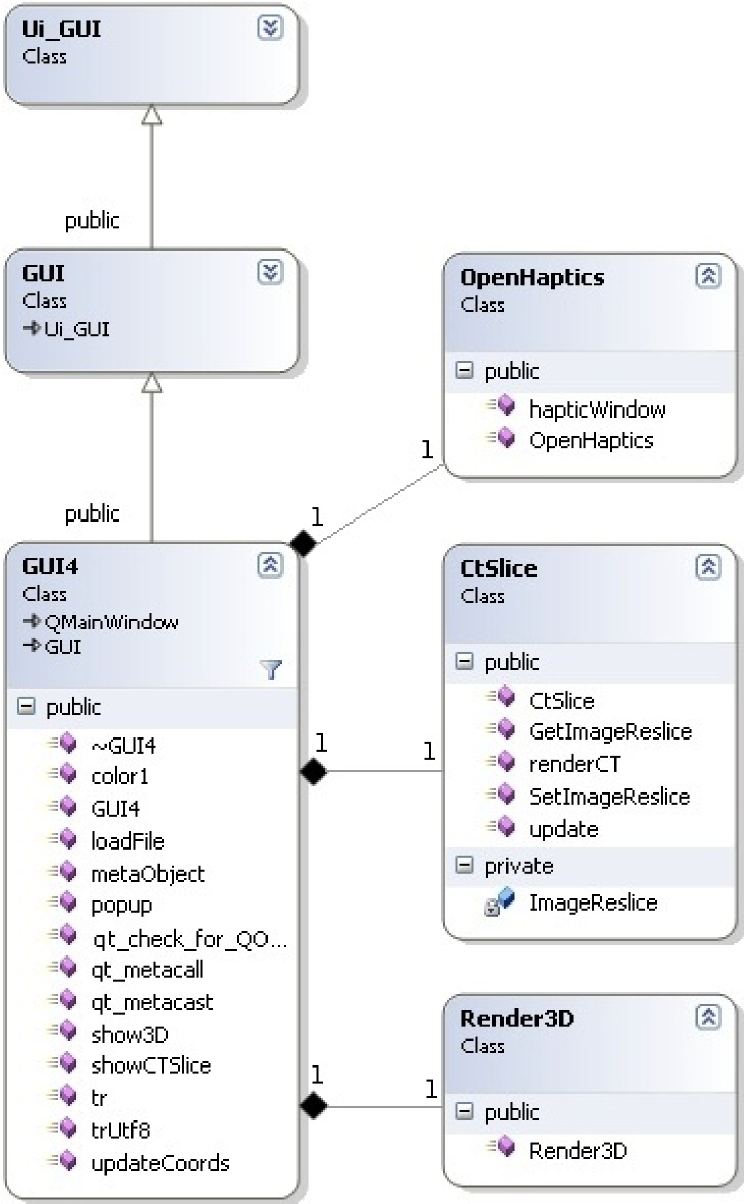


Figure 8.5.1: Class Diagram 2nd Sprint

Test ID - Name	T3 - Read position and integrate the device
Description	The application should be able to read the position with the Phantom Omni device in our application.
Criteria	The user moves the Phantom Omni device and shall see the pointer move accordingly.
Estimated hours	73
Actual hours	103
Results	Passed
Comments	Position is read in the application, but will be opened in another window since QuickHaptics does not support integration with Qt at this moment.

Test ID - Name	T4 - Implement force feedback
Description	The Phantom Omni Probe shall receive force feedback if you hit an object in the application.
Criteria	The user moves the Phantom Omni device and tries to “enter” inside the object. The user shall then perceive vibration in the Phantom Omni device.
Estimated hours	68
Actual hours	8
Results	Passed
Comments	The force feedback works really well.

Test ID - Name	T5 - Optimize the 3D rendering
Description	The application shall be able to surface render medical images without including what is “inside” the object.
Criteria	1. The user selects which medical images that will be 3D rendered. 2. The user shall then see a 3D model of the loaded medical image parts.
Estimated hours	64
Actual hours	52
Results	Passed
Comments	Works really good, and is much faster than the rendering we had in sprint 1.

8.7 Results

After the first sprint it was decided not to implement ultrasound image in the application. Now the application only have three views instead of four. In figure 8.7.1 starting from the left, we have the Haptic view, 3D image view, and CT slice view.

The CT slice view will be finished during the next sprint, because it is not possible to navigate it with the haptic device in real-time. 3D image rendering has improved a lot and is very fast now.

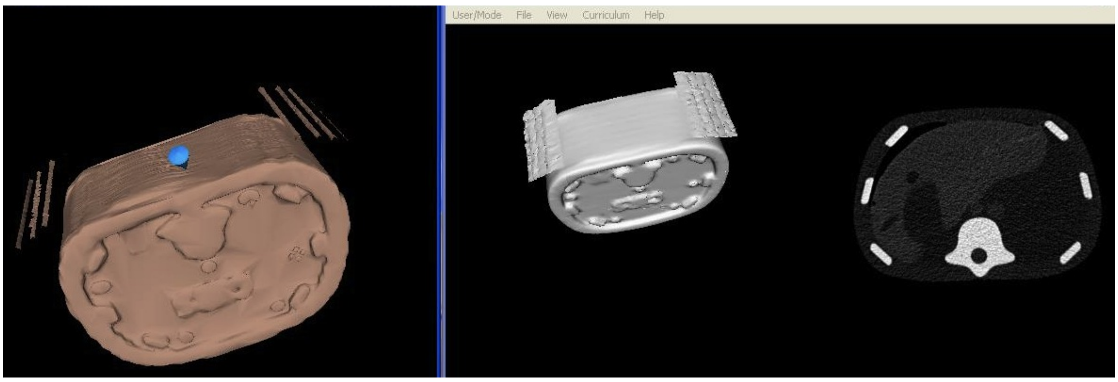


Figure 8.7.1: Application screenshot after the second sprint (part one)

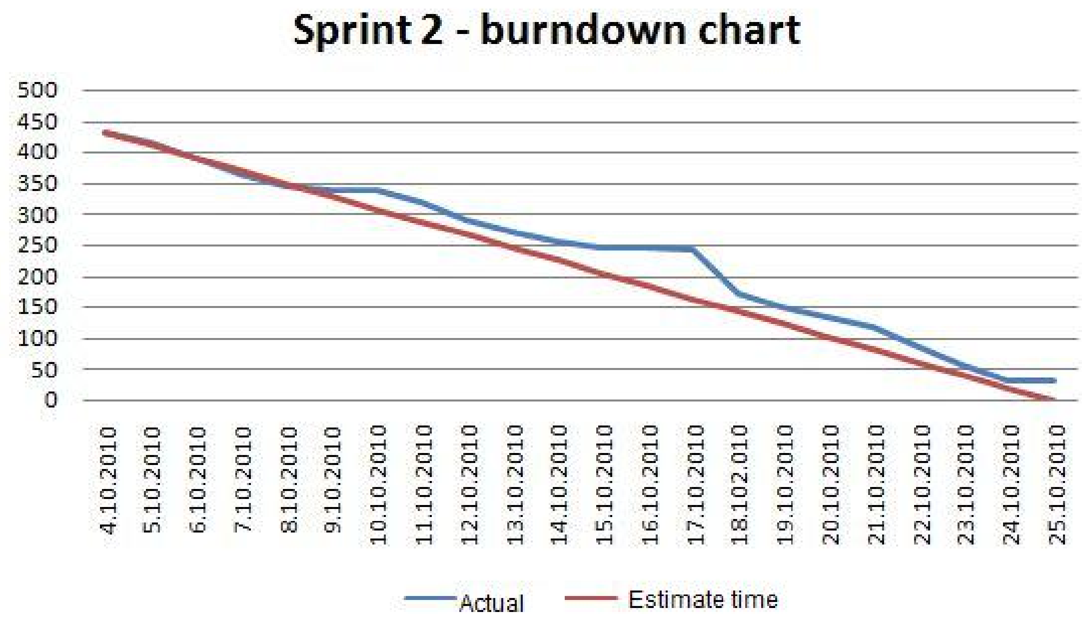


Figure 8.8.1: Burndown Chart Sprint 2

8.8 Evaluation

During this sprint we had much more knowledge on the different toolkits we are using, so we were able to focus more on development and specific implementation, and not so much on toolkit research. In this sprint there has been some large and complicated tasks, which has required a lot of time to solve. The different system components are not large in themselves, but integrating them with each other needed a lot of work, since the toolkits have different dependencies and constraints.

According to the burndown chart, this sprint estimation was better than in the first sprint. One item from sprint backlog (P14: View CT slice images) has not been finished yet. Thirty estimation units are needed to completeness and they are passed to the next sprint.

Chapter 9

Sprint 3

The third sprint meeting with customer was held at SINTEF Medical Technology on October 27th. In this customer meeting it was commented that the team was on good progress and the main task for this sprint will be to finish the realtime CT slice view. Also these items from the product backlog was picked: P5: Help access, P6: Load a case, P7: All platforms compatible, P11: Select from different curriculum and P12: Save a case. P7 and P6 are highly important items, while P5, P11 and P12 should be fairly quick to do, since for P5 and P11 there will only be a shell and interface developed so that the customer can include the relevant information themselves in the future. Save a case might take a bit more work.

In addition to these items, the team will work on the report because this is also the final sprint for this project. The full sprint backlog can be seen in the Appendix E. The sprint will end in a demo meeting for the customer.

9.1 Sprint Goal

The goal of this sprint is to finishing the work of extracting the correct CT slice based on the position of the haptic or mouse cursor, be able to load and save a case from the menu and to have cross-platform compatibility. There will also be some minor goals to make a simple interface or menu system for selecting curriculums and to access the help menu.

9.2 Sprint Backlog

ID	Priority	Description	Requirement	Hours est.	Hours used	Responsible
P6	High	Load a case	FR2	30	24	Facundo
P7	High	All platforms compatible	NR1	94	94	Facundo
P14	High	View CT slice images	NR3	30	30	Eivind
P12	Medium	Save a case	FR2	16	20	Facundo & Eivind
P11	Low	Select from different curriculum	FR6	3	4	Jelena
P5	Low	Help access	FR7	3	2	Dag Atle
P8	High	View 3D image and CT slice	FR3	196	192	Aldo, Facundo & Eivind

9.3 Sprint Planning

The sprint planning meeting with the customer was held at SINTEF Medical Technology on October 27th. Everyone except Jelena Petrovic and Reidar Brekken attended the meeting. Since product backlog items P13: Render ultrasound images, P10: Input parameter interface and P15: Realistic ultrasound images are removed from the backlog, as the customer suggested at the second sprint meeting, the item P8: View 3D image, CT slice and ultrasound image is also changed. This item include only the view of the 3D and CT slice image now. It was agreed that the main focus for this sprint should be on making the CT slice extraction in real-time. Making the application cross-platform compatible should also be prioritised, and to have an opportunity to load and save a case.

Load a case It is preferable for the user to be able to load a previous case as an OBJ file than having to make and render the medical images each time the case is to be used. Since the system is able to save the case as OBJ files, it should not be difficult to load that OBJ file and display it in the GUI. The group will not

estimate and allocate much time to this requirement, since it should be fairly easy compared to the other goals for this sprint.

All platform compatible Since people use different operating systems (Windows, Mac OS X, Linux), it will be preferable for the customer that the system can be used on all the different operating systems. This is also a high ranked requirement to solve in this sprint.

View CT slice (continued) In sprint 2 the team were able to view the correct CT scan slice based on coordinates passed from a 3D model in Qt, where the coordinates were updated based on the mouse coordinates when you moved the mouse. The algorithm is, though, far from perfect, so it needs improvement as mentioned in sprint 2 (see section 8.4).

View 3D image and CT slice In this sprint the problem will be to get the coordinates from the haptic device and get the correct CT slice based on the 3D model. This can be a challenge since the coordinates in the haptic world needs to correlate to the correct CT slice, even if the rendered 3D model is moved in any way (turned around, flipped over et cetera). This will be one of the major problems to work on during this sprint, since this is now one of the most important features in the system. If the haptic device is not used, then the user should be able to use the mouse on the 3D model and get correlating CT slices in the CT slice window on the right side in the application. Much of that is already done in sprint 2 (View CT slice), but it needs improvement.

Save a case Much of this is already developed, since the application needs to be able to save the 3D volume as an OBJ file to be able to make it work with the haptic device. What is left now is to make a simple interface to the user, so they can easily do this themselves. This will save time so they do not have to render the image files the next time they plan to use the same case.

Select from different curriculum This is very low prioritised and is basically to make a simple menu item to establish an infrastructure for future expansion where the customer can add different curriculum information for cases.

Help access This is also very low prioritised, but is mainly to have an infrastructure for future expansion for a help section in the application if users do

not know what to do. But there will also be written a user documentation with instructions on how to use the system.

9.4 Actual Process

Load a case The team generalized the image support by supporting DICOM files, MetaImage files and OBJ files, so when the user chooses "Load a case" in the menu, the user gets an option to choose the image files that will be used. DICOM files and MetaImage files requires volume rendering. OBJ files is already a 3D surface and requires no rendering.

All platform compatible In order to make the application cross-platform compatible, CMake (see section 2.3.7) have to be used. The first thing that had to be done was to make a CMakeLists.txt file which informs the CMake program where all the different libraries and packages in ITK, VTK, Qt and OpenHaptics are. Then CMake is used to generate project files based on chosen IDE and platform, and thereafter the project is built. This task was very time consuming since it was not easy to make the CMakeLists file which uses another programming language no one in the group have used before. This solution was chosen over making installation packages, firstly because making three different installation packages would have taken even longer time, but more importantly the program is still a prototype and needs more functionality like ultrasound simulation before it is finished. In other words the team found it not necessary to make installation packages when the program are missing some core functionality.

View CT slice (continued) The group were able to fix the algorithm problem by changing the calculation of the center parameters, so it does not anymore pick only the top and middle slices.

View 3D image and CT slice This was the main challenge in this sprint because getting the coordinates from the haptic device and translate them into the correct CT slice is not an easy task. The team started first to translate the world coordinates in a 3D volume to the CT slice window to add mouse navigation support to the application. This was mostly done in the View CT slice part of sprint 2 and sprint 3. When that was done, the team had to do the same with the haptic 3D volume. This was a huge challenge because the OpenHaptics system uses another coordinate system, so again the team had to synchronize that coordinate system with the CT slice coordinate system. It was also decided to use the world

coordinates from the QuickHaptics system to the CT slice window. The algorithm to view CT slices was the same as in the 3D volume to CT slices, so nothing had to be changed there. A simple check were also added to make sure that the CT slice only updates when the user interacts with the 3D model itself.

Save a case Saving a case is really now just “Export surface”. This means that the system now exports and saves the surface of the 3D model in an OBJ file and stores that in the folder the user selects. This took a bit longer time than expected because we needed a good compression algorithm because if the file is too big it takes much longer time to load and navigate with. There were two compression algorithm included in VTK, ProDecimate and QuadricClustering. The first one created a larger file than QuadricClustering, but had a much smoother 3D object, and we noticed that even though the file was larger it did not effect the loading time substantially. So in the end we chose ProDecimate.

The other non-expected problem we had, was when we saved an OBJ file with the OBJWriter class in VTK and thereafter tried to load it again, VTK automatically saved a header file, and because of that header file we could not load the OBJ file back into our program. So we decided to modify the OBJWriter class so it did not write the header file. This class is called vtkOBJModifyExport in our program.

Select from different curriculum This was not fully implemented since the customer said that this has low priority, so the only thing that is implemented here is the interface and a simple GUI shell. This makes it reasonably easy for the customer to add the curriculum information themselves in the system.

Help access Like the curriculum selection, this part is not fully implemented in the system. Only the interface and a simple GUI shell are created, so it is left for in the future to add an user manual. We do, however, have created a short user documentation in Chapter 10.

9.5 Implementation

Figure 9.5.1 shows the UML class diagram for Sprint 3. This was our largest Sprint. The issues with synchronizing the CT slices and 3D model was the largest task during this sprint and posted a huge challenge for the team. There was no easy way to use coordinates strictly from the model to get the correct CT slice, so we decided to use a solution with world coordinates. After this task, the rest of the product

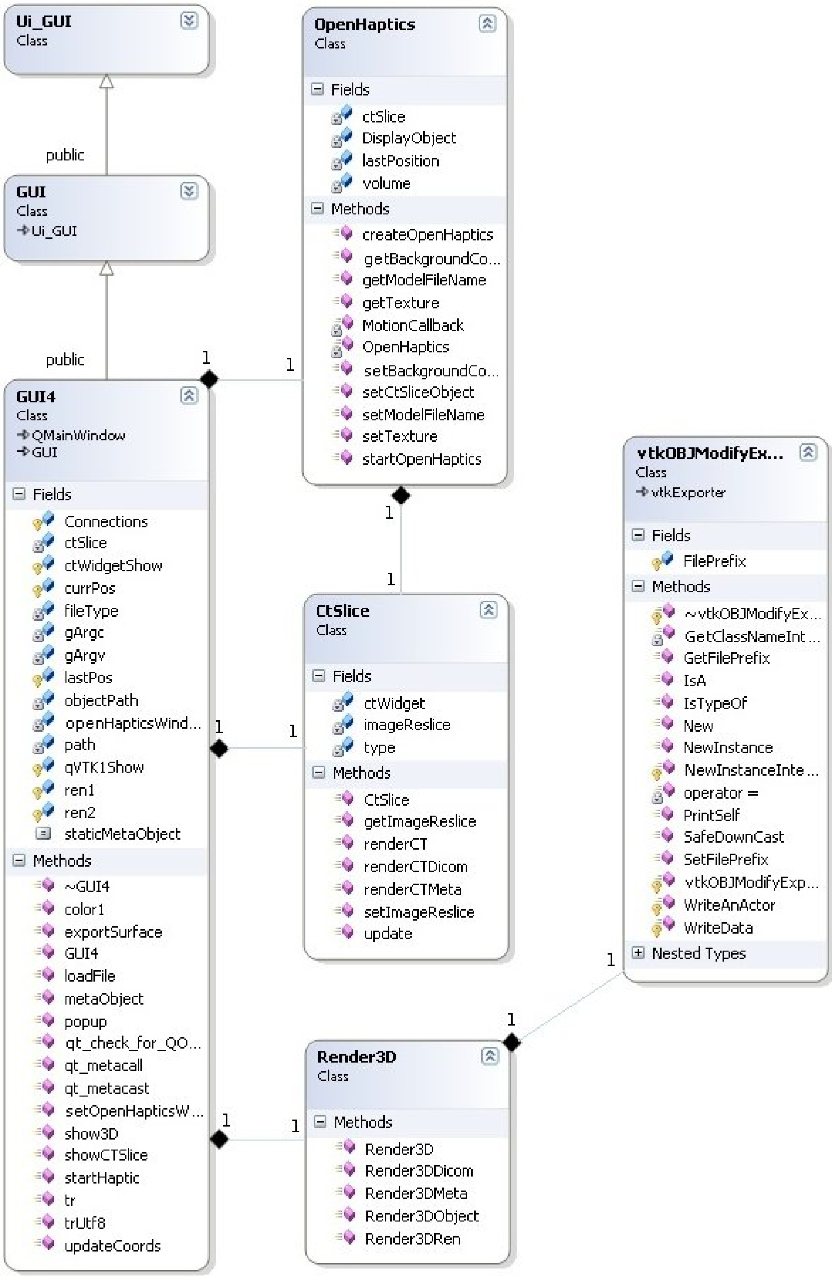


Figure 9.5.1: UML Class Diagram Sprint 3

backlog items were done in somewhat rapid succession. Making the application all platform compatible took quite some time, though, and in product backlog item save a case we got two unexpected problems with the compression algorithms and with the OBJWriter class in VTK. Since the team and the customer agreed on focus on the high and medium backlog items, the team decided to simply add an interface for the lower priority items so it can easily be included in the future. In retrospect, the team sees that this was a good thing since at this point there would not be any time to integrate these items. The remainder of the development time will be used to finish the report and to prepare for the final presentation at the end.

9.6 Testing

Testing at the end of sprint 3 is pretty much testing to see if the whole system works together, as well as the previously stated test methods of testing each individual component to see if they work in the system. The testing methods have not changed since the last sprint, but there will be a large system test in the end of the sprint to do a final check on how the system works at this point. Testing each component along with the others is also important now, since having a working and good system is vital. The following test cases are what the team deemed as the most important parts of this sprint.

Test ID - Name	T6 - Generate image slices
Description	The application should be able to view and render image slices in realtime.
Criteria	The user navigates in the 3D model. Then, the application generates an image slice based on the position of the cursor in the 3D model.
Estimated hours	72
Actual hours	102
Results	Passed
Comments	This works in both haptic view and regular 3D view in Qt.

Test ID - Name	T7 - Load a case
Description	The user should be able to load an old case.
Criteria	The user chooses “File” in the menu and selects “Load case”. Then, the user shall be able to import an old case stored on the computer.
Estimated hours	30
Actual hours	24
Results	Passed
Comments	The user can not use the CT slice view if an OBJ file is loaded. This is because an OBJ file does not have anything inside since it is just a 3D surface.

Test ID - Name	T8 - Save a case
Description	The user should be able to save the 3D model it has created.
Criteria	The user chooses "File" in the menu and selects "Export surface". Then, the user is able to select where the 3D model is going to be stored on the computer as an OBJ file.
Estimated hours	16
Actual hours	20
Results	Passed
Comments	None

Test ID - Name	T9 - All platforms compatible
Description	The application should run on Windows, Mac and Linux.
Criteria	The administrator uses CMake to build the project on the desired platform. Then the administrator builds and starts the program.
Estimated hours	94
Actual hours	94
Results	Passed
Comments	This is more a way to build the program on the major platforms.

9.7 Results

The selected backlog items and the goals for this sprint were finished. There is not much of a visible change in the GUI, but the user can now extract the CT slices based on the position of the haptic or mouse cursor, and functionality like load and save a case is implemented.

From now on the main focus will be to work on the report and make the final presentation. Figure 9.7.1 shows a screenshot of the finished system.

The feedback from the customer after the final sprint meeting was positive. They were impressed that the user could load three different image files and get a fast and good-looking 3D model with force feedback functionality in all of them. Especially the force feedback feature is vital because then the medical personnel can focus on the problem without looking at the 3D model. They also liked the CT slice view, but the algorithm may need some refinement for transformation of the 3D object.

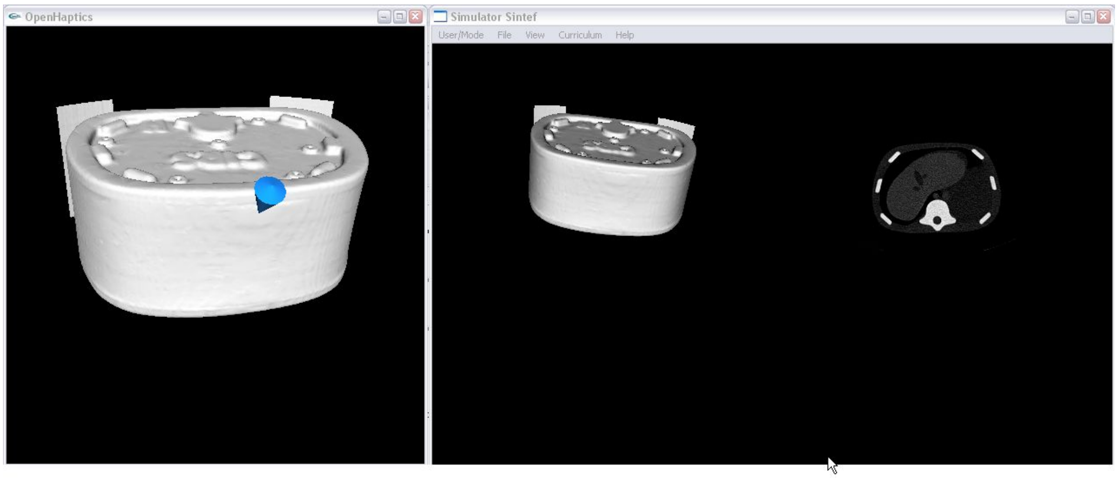


Figure 9.7.1: Screenshot of the finished system

The CT slice itself needed some adjustment since the colour range was too low, so many slices became too bright. This was fixed momentarily.

9.8 Evaluation

According to the burndown chart shown in the figure 9.8.1 the most items which are not so time consuming are left for the last sprint. This is the reason why the actual time spent in this sprint is not that high as it was estimated. This allowed the team to spend more time on the report and the finishing touch of the product. The goal on the beginning of the project was to avoid the stress when the deadline is getting close, and the team succeed to some extent in this. The work is finished according to schedule and the team proceeded with the final report writing and to prepare the final presentation.

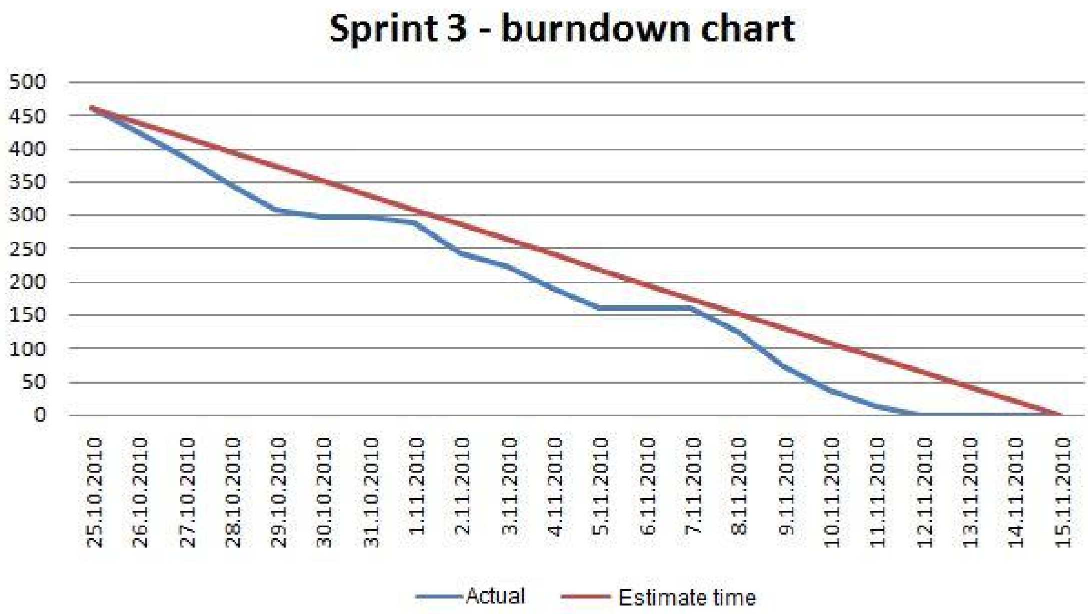


Figure 9.8.1: Burndown Chart Sprint 3

Chapter 10

User Documentation

This chapter describes how to use the application and gives a short description on how to install the necessary files and libraries to make the system run on a computer.

10.1 System requirements

Operating systems The program is multi-platform compatible and only uses open-source libraries that works on the three major platforms, Mac OS X, Windows and Linux. However, the program has been tested on Microsoft Windows 7 and Windows XP, and these operating systems are therefore highly recommended.

Software requirements The program requires the following software:

- ITK 3.20.0
- VTK 5.6.1
- OpenHaptics 3.0
- CMake 2.8.3
- Qt 4.5.2

10.2 Installation guide

To get the application running on the different platforms, the project has to be deployed on that system. In order to make it easier for the user, we have prepared a CMakeLists.txt file, so that the administrator can generate the project and run the project accordingly. Here is a short step-by-step guide:

1. Download and install CMake from http://www.cmake.org/cmake/resources/software.html.
2. Install ITK, VTK and Qt as described in the ITK web page http://itk.org/ITK/help/tutorials.html.
3. Download and install OpenHaptics from http://www.sensable.com/products-openhaptics-toolkit.htm.
4. Copy our source files and the CMakeLists.txt file to a source folder.
5. Run CMake and select the source folder where you copied all the files using "Browse source."
6. Select a new folder where you want to build the project files using "Browse Build."
7. Click on "Configure" and wait few minutes.
8. Click on "Generate".
9. All the project files will be in the build folder and can thereafter be build accordingly.

10.3 User manual

10.3.1 Graphical User Interface

A window should open when the program is successfully started. From the window, menus and empty views panes are shown on the user's screen (see figure 10.3.1). The menu bar at the top, provides access to the different menus to execute a given task. The view panes are found below the menu bar and are aligned in a grid form providing the user with the various views of the model. In the following sections, we will describe how you can intuitively use these components to accomplish different tasks to achieve greater accessibility, lower cognitive load and higher productivity by spending less time on figuring out how to achieve a task.

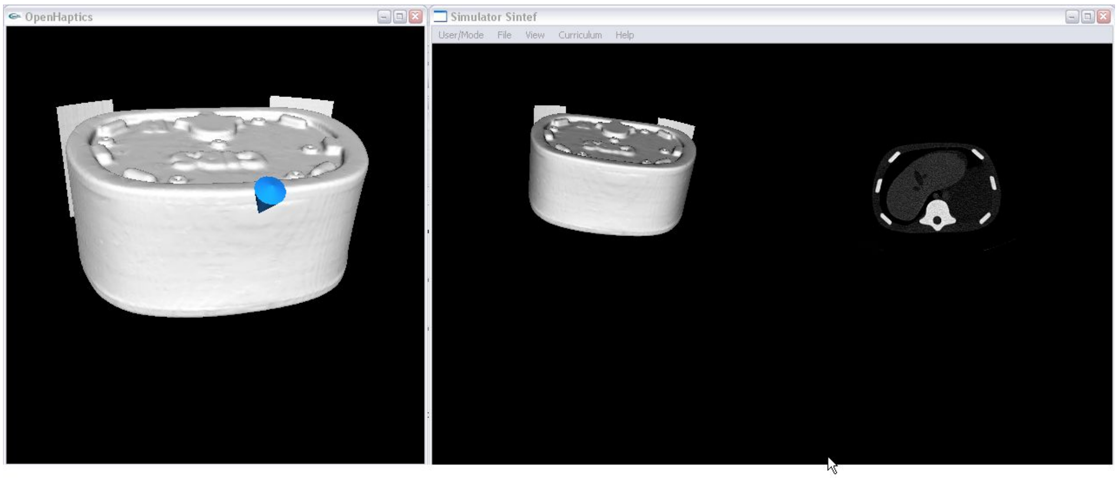


Figure 10.3.1: Screenshot of the main window showing the various views and GUI components

View Menu As is shown in figure 10.3.1, the application has several views available accessible from the menu "View". Starting from the right you can see:

- **Axial View**: this is where the CT slice is shown;
- **3D View**: this is where the user can interact with the 3D model using the mouse;
- **3D OpenHaptics View**: this is where the user can "touch and feel" the 3D model using the haptic device.

File Menu

Loading a new case

1. Start the application;
2. On the menu bar, click on "File menu" (see figure 10.3.2).

Export/Save a surface

1. Load a case
2. Save the 3D model at your chosen destination as an OBJ file by clicking on File > Export surface (see figure 10.3.3).

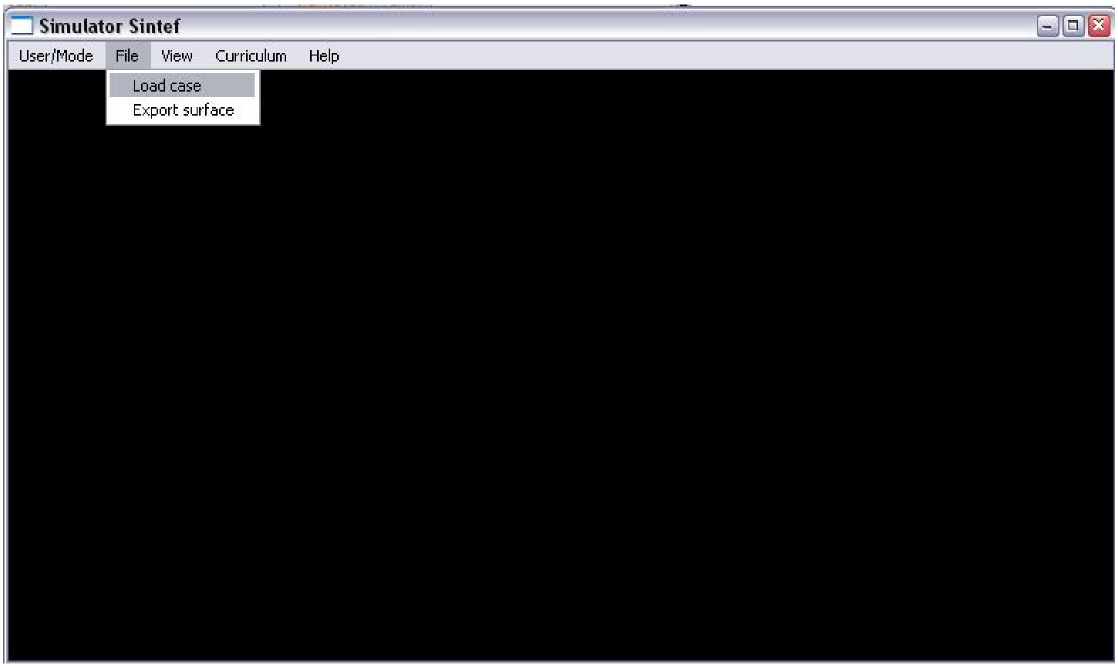


Figure 10.3.2: Loading a new case screenshot

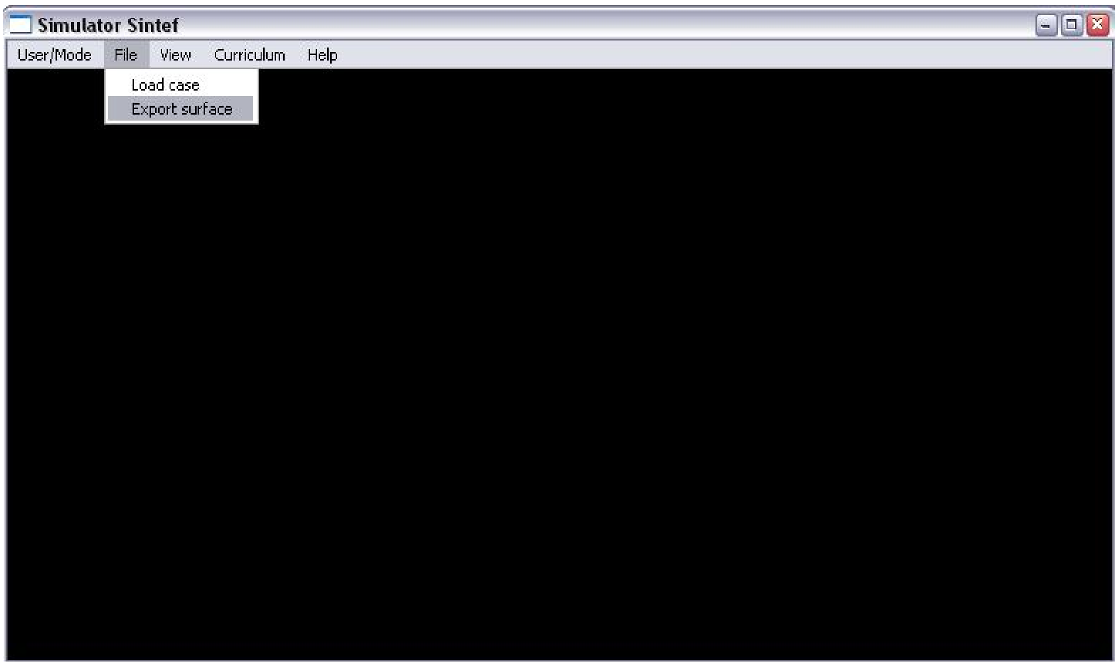


Figure 10.3.3: Exporting a case screenshot

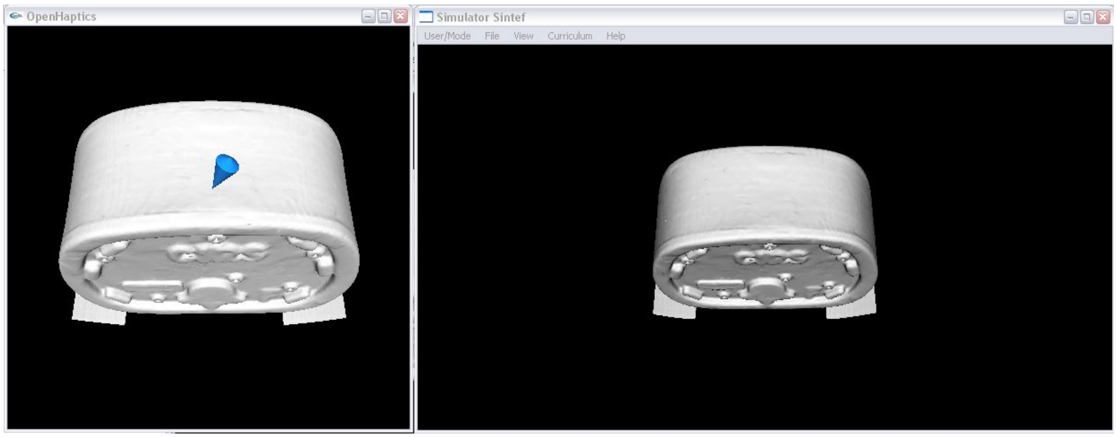


Figure 10.3.4: Haptic and 3D model views

10.3.2 3D Model Interaction

For interaction with the 3D model on the screen (see figure 10.3.4), the user should handle the stylus of the Phantom Omni device and explore the 3D model on the display as they would if they were using a pen to touch an object. By moving the stylus one can feel objects (through force feedback) by the direction and magnitude of the opposing force synthesised by the device. The user can also chose to interact with the mouse, if the haptic device is not present.

Chapter 11

Evaluation and Conclusion

This chapter contains the evaluation of the whole project. The project task will be explained in more detail. The challenges and doubts the team coped with along the way will be presented in this chapter which includes relations with customer, supervisors and internal relations among team members. The evaluation in the development process will also be mentioned, but it was briefly described in Sprint cycle chapters 7, 8, 9. The final results are presented and everything is sum up in the conclusion.

11.1 Project task

After the first customer meeting it was obvious that the team got quite a challenging task. All the group members were not familiar with any of the toolkits which were suggested by the customer to use in the project. At first it seemed like a very big and demanding task. Later on, after the team has done some preliminary studies and tried most of the toolkits, it was easier to break down the task. We found out that the most of the toolkits provide a lot of already implemented solutions.

The choice of an agile development framework was essential for getting the product as close to the customer expectations, as possible. One of the main objectives of this project was to develop the simulation of ultrasound based on medical images. On the beginning of the second sprint, the requirements were changed, since the customer could not finish the ultrasound algorithm and the product backlog was updated according to that. Due to the new situation, the scope of the project was changed, and one of the main objectives was removed. This led to reorganisation of the project plan.

11.2 Customer relation

The customer and sponsor of this project was SINTEF Medical Technology. The choice of the programming language and most of the toolkits used in this project were defined by SINTEF. They have many years of experience in this field, which was very helpful and made them more advisers of the team, not only customers. They also showed great understanding about the workload and report writing.

The customer meetings was usually short and efficient. The customer supported us in every decision we made and was satisfied with each sprint demo. During the project the team had some technicalities to deal with like borrowing the haptic device from the company, find a secure place to store it and buy a FireWire adapter so the haptic device can work on more than one computer. The customer was always willing to help in solving these problems in the shortest time possible. It was a pleasure to collaborate with them.

11.3 Supervisors

The team was provided with two supervisors who were committed and willing to help anytime. We had a meeting every week where they evaluated and provided feedback for our documentation and our weekly status report. Meeting agenda, minutes from last supervisor meeting, time sheet with estimated and actual working hours and weekly status report were written by the secretary every week, and were sent to the supervisors the day before the meeting. Sprint burn down chart and detailed sprint backlogs were also provided in the end of each sprint, so the supervisors could keep track of our working progress. They were, also, very interested in the topic and the application was demonstrated in these meetings many times, as well.

We experienced the supervisors to have good knowledge on the Scrum framework and the report writing, which were invaluable resources in this project for the team.

Three weeks before the end of the project, main supervisor was changed and we got a substitute. It was unfortunate since the original supervisor had very good knowledge about us and our progress. We collaborate with the main supervisor from the beginning and it was not easy to finish some tasks that he have been suggested without his assistance. The bad timing makes this situation even worse, because we had so many doubts about report writing in the end of the project. In the end, we did managed to finish the report.

11.4 Team Conflicts

The structure of our team was heterogeneous. The team consisted of four international students and two Norwegian students. Since we decided in the beginning that we will work in the same room (we had work sessions of 6-8 hours three times a week), we spent much time together. This created a good group dynamic for those who attended and we easier got to know each other. Learning a lot about differences in the culture, habits, language, politics in our countries was the essential part of our meetings, as well. This make work on this project memorizable and a nice experience for all of us.

We had some major team conflicts during the project. It was mostly about the way of working. The whole team agreed in the beginning that we will meet three times a week to have regularly Scrum meetings and for those who wanted, people can sit together and work on the project after the meetings. However, a team member had problems to align to this since he had both personal problems and technical problems with his laptop. And when a person comes late on the Scrum meetings, the other team members need to spend time and energy explaining the discussion one more time. Everyone can be late to a meeting sometimes, but when it starts happen regularly, the other team members agreed it was needed to consult the main supervisor about that problem, since the group unfortunatly could not solve this problem among themselves. However, after several consultations these problems were solved and we all agreed to continue the original plan of having regular Scrum meetings. After that the team returned to work together and everyone arrived at time for the meetings. Taking this up with the supervisors really had an impact, and looking back, we should have gone to them earlier because the conflict just grew bigger and bigger the more we postponed the conflict.

11.5 Development Process

Scrum as an agile method was used in this project. In the beginning we had an overall plan to have short and small sprints which we loosely tried to aim for four sprints of fixed duration of two weeks. It was difficult to cope with the new technologies and this problem continued mostly during the whole project, but of course we did improve and learned a lot, but all of the different toolkits are very large and advanced, and it will take much more than 3 months to have a solid expertise in them. Since so many problems had appeared during the whole development process, it was necessary to reinvent new approaches, find another solutions and explore different algorithms. Due to these reasons, it was better to expand the duration of some of the sprints, especially after the ultrasound

simulation was postponed from our project. After the first sprint, which duration was two weeks, the second and the third sprint were three weeks long. Each sprint started with a sprint planning meeting and ended with a sprint demo meeting. The team decided to organise both in one session. So, each customer meeting consisted of two parts: demonstration of the previous sprint and planning the next one.

During the development process it was mandatory to install all the different frameworks and toolkits, which took much time for each team member. When it came to integrating all the different technologies, it was very challenging to solve. But after doing some research and consulting with the customer, the team found a way to solve the integration problems and in the end delivered a product the customer seemed satisfied with.

11.6 Results

In the end of the project the team is pleased with the results. Most of the high and medium prioritized requirements are included in the application. Some product backlog items, which are related with ultrasound simulator implementation, were skipped. The interface for this was, though, made so it can be implemented later when they finish with the development of the ultrasound algorithm. In all, the final results fulfilled expectations of the customer.

Another big task was the report writing. We decided to try to divided the chapters, so everyone gets some experience in writing reports. Our supervisors were very good at focusing on the report writing at every meeting, so we managed to write on the report regularly to prevent to much report work in the end. Misfortune was that the main supervisor was not present in the end of the project, so the final version of the report was mainly written and reviewed by ourselves. We decided to divide the document for revision also. Assistant supervisor suggested this method, so we minimised the risk of the grammar and spelling mistakes as much as we can.

As it is shown in figure 11.6.1, the team almost spent as many hours on particular parts of the project as it was planned. The whole workload looks satisfactorily according to the bar chart.

The work effort spent on the project is presented with the line chart shown in figure 11.6.2. According to that, there were several weeks when the team relaxed and worked less, and it was usually due to some external conditions, for example the customer was not available the week after the first sprint, so we decided to work on the report for pre-delievery, but there were a limit of how much work on the report could be done at this time. Also, the team have been working longer in the end of the each sprint and when the deadlines was close. This can be hard

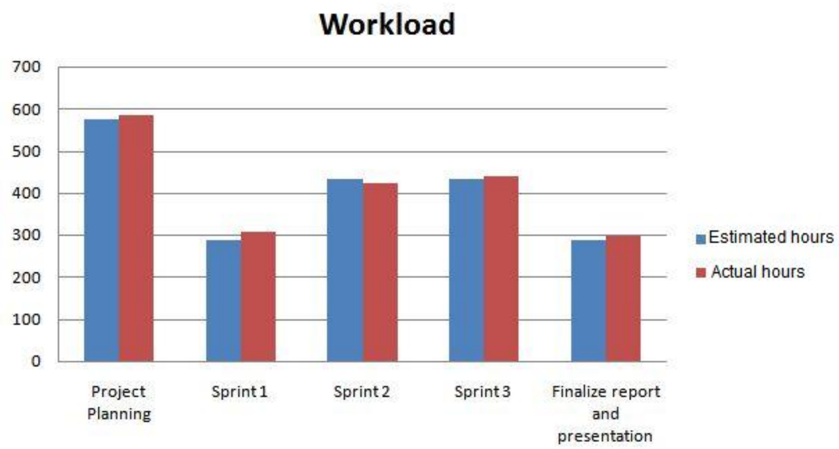


Figure 11.6.1: Actual vs. estimated hours spent on the planned parts of the project

to prevent since when we have finished solving one problem there almost always seemed to arrive a new problem to solve. However, we are satisfied with the work done, dispute the fact the correctness of the estimation per week was a bit variable.

11.7 Conclusion

We all think it was valuable to work on a real project with customers who had a real complex problem. It seemed like an impossible mission at the beginning, but actually turned into a thrilling experience and a valuable product for the customer.

We learned many interesting things in this project. The fact that all the involved technologies were unknown to us made us gain new knowledge and feel motivated most of the time. Exploring new fields and trying new toolkits are always fun and challenging. Also, the fact that the application may be used in the future for medical training was very motivating.

As an agile methodology, Scrum was also interesting to use since it was a new experience for all team members. Also, looking back on the many tasks that has changed and on some of the modified team members roles, it was really good to use an agile method to more easily handle these issues.

Figure 11.6.2: Actual vs. estimated hours per week

Glossary

3D Model is a type of representation of image data which makes it looks like it is in three dimensions, storing data in x, y and z dimensions.

API (Application Programming Interface) is an interface for linking the application/libraries with other applications. It usually contains procedures and methods for converting imported or exported data to data either the application or the other applications can use.

GUI (Graphical User Interface) is an interface which allows the user interacts with the system. This can be a window with buttons and sliders, or a simple command line interface.

IGS (Image Guided Surgery) is the general term for the surgical procedures where the surgeon uses medical imaging to guide the surgical instruments through a patient. A technique is mainly used on brain tumors, but it is now being frequently and widely used in many different fields of medicine, as well.

LaTeX is a markup language for document preparation. It is especially used for making more professional looking pdf files.

MATLAB is a numerical computing environment developed by MathWorks. It allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, and Fortran.

MetaImage is an image header that is expected to have extension: ".mha" or ".mhd". It usually points to a raw binary data of the image that typically have extension ".raw".

MRI (Magnetic resonance imaging) is a medical imaging technique in the field of radiology to get detailed images of different parts of the body. It gives a good contrast for brain, muscles, heart and cancer conditions.

OpenGL (Open Graphics Library) is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics. It was developed by Silicon Graphics Inc.

OsiriX is a processing application for viewing DICOM images on Mac OS X.

Phantom is the haptic device made by Sensable, who also made the OpenHaptics API that are used in this project.

Bibliography

[1] Sintef Web Site. http://www.sintef.no/Home/About-us/. Accessed on 02/11/2010.

[2] ScanTrainer Web Site. http://scantrainer.com. Accessed on 02/11/2010.

[3] N. Parekh. The Waterfall Model Explained. http://www.buzzle.com/editorials/1-5-2005-63768.asp. Accessed on 02/11/2010.

[4] Methods & Tools. http://www.methodsandtools.com/ . Accessed on 02/11/2010.

[5] Mountain Goat Software. http://www.mountaingoatsoftware.com/topics/scrum. Accessed on 02/11/2010.

[6] Medline Plus. http://www.nlm.nih.gov/medlineplus/diagnosticimaging.html. Accessed on 02/11/2010.

[7] Net Doctor. http://www.netdoctor.co.uk/health_advice/examinations/ctgeneral.htm. Accessed on 02/11/2010.

[8] Ragiologyinfo.net.http://www.radiologyinfo.org/en/info.cfm?pg=bodyct. Accessed on 02/11/2010.

[9] Radiologyinfo.net. http://www.radiologyinfo.org/en/info.cfm?pg=genus. Accessed on 02/11/2010.

[10] G. Robles-De-La-Torre. "International Society for Haptics: Haptic technology, an animated explanation". Isfh.org. Accessed on 26/02/2010.

[11] Sensable. What is haptics. Accessed on 02/11/2010.

[12] B., Stroustrup. Bjarne Stroustrup's FAQ. http://www2.research.att.com/~bs/bs_faq.html. Accessed on 02/11/2010.

[13] H. Schildt. “C++ The Complete Reference Third Edition”. Osborne McGraw-Hill. ISBN 978-0078824760.

[14] Sensable. Open Haptics Programmer’s guide.. Accessed on 02/11/2010.

[15] CMake Web System. http://www.cmake.org/. Accessed on 02/11/2010.

[16] VTK Web Site.http://vtk.org/. Accessed on 02/11/2010.

[17] ITK Web Site.http://itk.org/. Accessed on 02/11/2010.

[18] Diagnostic Imaging.http://www.nlm.nih.gov/medlineplus/diagnosticimaging.html. Accessed on 02/11/2010.

[19] Microsoft MSDN.http://msdn.microsoft.com/en-us/vstudio/default.aspx. Accessed on 02/11/2010.

[20] Nokia. Qt Web Site.http://qt.nokia.com/products/. Accessed on 02/11/2010.

[21] IEEE Recommended Practice for Software Requirements Specifications. http://ieeexplore.ieee.org/iel4/5841/15571/00720574.pdf

[22] IEEE Recommended Practice for Software Design Descriptions.http://ece.iut.ac.ir/Faculty/davarpanah/SE/ieee1016-1.pdf. Accessed on 02/11/2010.

[23] Eclipse Web Site. http://www.eclipse.org/. Accessed on 02/11/2010.

[24] NetBeans. http://netbeans.org/. Accessed on 02/11/2010.

[25] P. Yushkevich, J. Piven H. Hazlett, R. Smith et al., “User-guided 3D active contour segmentaion of anatomical structures: Significantly improved efficiency and reliability.” Neuroimage, 2006

[26] J. Cebral, R. Lohner, “From medical images to automatically accurate finite element grids” Int J. Numerical Methods in Engineering; 2001

[27] N. Ezqueera, I. Navazo, T. Morris and E. Monclus: “Graphics, vision and visualization in medical imaging. A state of the art report” Milano, Italy. Eurographics 1999

[28] W. Schroeder, K. Martin, and B. Lorensen, The Visualization Toolkit – 2nd Edition, Prentice Hall, Upper Saddle River, NJ, 1998.

[29] Customer Driven Project Web Site. http://www.idi.ntnu.no/emner/tdt4290/. Accessed on 02/11/2010.

[30] Chai3D Web Site. http://www.chai3d.org/. Accessed on 02/11/2010.

[31] Czech Technical University in Prague. Department of Mechanics Web Site. Accessed on 02/11/2010.

[32] Royriggs.com. http://www.royriggs.com/obj.html. Accessed on 02/11/2010.

[33] H. K. Cakmak and U. Kuhnapfel, The Karlsruhe endoscopic surgery trainer for minimally invasive surgery, Paris, June 1999.

[34] M. Bro-Nielsen, D. Helfrick, B. Glass, X. Zeng and H. Connacher, VR simulation of abdominal trauma surgery, In proceedings of medicine meets virtual reality, San Diego CA, 1998.

[35] Osirix Viewer Web Site. http://www.osirix-viewer.com. Accessed on 02/11/2010.

[36] Y. Bar-Cohen, C. Mavroidis, C. Pfeiffer, C. Culbert, D. Magruder, Haptic Interfaces, 1999.

[37] Via technologies, Inc. http://www.via.com.tw/en/index.jsp. Accessed on 02/11/2010.

[38] H. Toetenel, R. Spelberg, G. Bandini, Parametric Verification of the IEEE 1394a Root Contention Protocol using LPMC, 2000.

[39] Sensable Web Site. http://sensable.com/. Accessed on 02/11/2010.

[40] D. Ruspini, K. Kolarov, O. Khatib, The Haptic Display of Complex Graphical Environments, Proc. of ACM SIGGRAPH, 1997.

[41] G. Fitzmaurice, Bricks: Laying the foundations for graspable user interfaces, ACM Press, 1995.

[42] F. Lindseth, T. Hernes, J. Kaspersen, T. Langø, J. Bang, G. Tangen, T. Selbekk, E. Sjølie, H. Brekken, S. Svee, J. Otto, N. Andersen, Navigation Accuracy.

[43] M. Zhao, J. Tian, X. Zhu, J. Xue, Z. Cheng, H. Zhao, Design and implementation of a C++ toolkit for integrated medical image processing and analyzing, Proc. Of SPIE: Medical Imaging, 2004.

[44] Fourth Dimension. Wikipedia. http://en.wikipedia.org/wiki/Fourth_dimension. Accessed on 02/11/2010.

[45] OSGi framework. Wikipedia. http://en.wikipedia.org/wiki/OSGi. Accessed on 02/11/2010.

[46] K. Mühler, C. Tietjen, F. Ritter, B. Preim, The Medical Exploration Toolkit: An Efficient Support for Visual Computing in Surgical Planning and Training.

[47] D.Maleike, M. Fabel, R. Tetzlaff, H. Von Tengg-kobligk, A. Hans-peter Meinzer, I. Wolf, Guided Deformable.

[48] N.Toussaint, M. Sermesant, P. Fillard, A VTK Extension for Spatiotemporal Data Synchronization, Visualization and Management Release, 2007.

[49] W. Schroeder, K. Martin, W. Lorensen, The Design and Implementation of an Object-Oriented Toolkit for 3D Graphics and Visualization, 1996.

[50] TheLabs.com. http://www.the-labs.com/Blender/3DS-details.html. Accessed on 02/11/2010.

[51] Chai3D.org. http://www.chai3d.org/concept2.html. Accessed on 02/11/2010.

[52] E. Bänsch, K. Mikula, "Adaptivity in 3D Image Processing", COMPUTING AND VISUALIZATION IN SCIENCE, 2001.

Appendix A

Original Project Description

D7 Sintef Medical Technology – Frank Lindseth

Title: Development of a real-time training simulator for medical ultrasound imaging
Customer: SINTEF
Address: SINTEF
Dept Medical Technology
Medisinsk teknisk forskningssenter, Olav Kyrres g. 9, 7489 Trondheim

The introduction of small and inexpensive ultrasound scanners will increase the use of ultrasound within a variety of medical fields. A potential consequence is that users with little experience in acquisition and interpretation of ultrasound images will be operating these scanners. These users will need efficient training. In several clinical applications it is not feasible to use actual patients for large scale training, such as in trauma where ultrasound can be used for detecting internal bleeding. Use of ultrasound training simulators is recognized as an interesting alternative for learning basic skills. An ultrasound simulator uses a model of human anatomy to generate realistic ultrasound images in real-time based on user input.

Methods for ultrasound simulation have been developed in collaboration between several SINTEF departments, NTNU, St. Olav's Hospital and a commercial company. To explore new potential areas and applications for the simulation methods, the objective for this project is to **develop (and implement) a real-time ultrasound simulator as a desktop application, integrating the simulation methods with a force-feedback haptic device.** Because real-time simulation of realistic ultrasound images is computationally demanding, an interesting task would be to investigate the potential of using GPU for efficient processing. In addition to the haptic device, efficient GUIs and visualization of anatomy and images are important. The simulator could serve as a framework for research into new simulation methods, and for demonstration and potential later commercialization in cooperation with other partners.

Contact details:
Name: Frank Lindseth
Mobile: 928 09 372
E-mail: frank.lindseth<#>sintef.no

Navn: Reidar Brekken
Mobile: 930 59 651
E-mail: reidar.brekken<#>sintef.no

Fax: 930 70 800

Appendix B

Templates

B.1 Weekly Status Report

Weekly status report, week No, group 7

Project: Group 7, Sintef Medical Technology

Prepared by: Jelena Petrović

Summary

Activities planned for last week

Activities accomplished last week

Activities planned for next week

New issues

Old issues

Timesheet

B.2 Supervisor Meeting Agenda

Agenda - supervisor meeting, DD.MM.YYYY

Project: Group 7, Sintef Medical Technology

Prepared by: Jelena Petrović

Invited Supervisor: Meng Zhu

Assistant Supervisor: Tobias B. Iversen

Team:

- Facundo Fortuny,
- Eivind Bjørkelund,
- Jelena Petrović,
- Dag Atle Midttømme,
- Aldo Doronzo,
- Simon Takite.

Place and time: ITV 054, DD.MM.YYYY at 14:00-15:00

Topics:

- Approval of agenda
- Approval of the status report
- Documentation review
- Other issues

B.3 Supervisor Meeting Minutes

Meeting minutes - supervisor meeting, DD.MM.YYYY

Project: Group 7, Sintef Medical Technology

Prepared by: Jelena Petrović

Invited Supervisor: Meng Zhu

Assistant Supervisor: Tobias B. Iversen

Team:

- Facundo Fortuny,
- Eivind Bjørkelund,
- Jelena Petrović,
- Dag Atle Midttømme,
- Aldo Doronzo,
- Simon Takite.

Place and time: ITV 054, DD.MM.YYYY at 14:00-15:00

Notes

Appendix C

Risks

ID	Type of risk	Probability (1-10)	Damage (1-10)	Risk (1-100)	Strategy	Responsibility
R1	Illness	6	3	18	Everyone has to notify the others about their illness, and we will have to try not to be depended on one single person.	Everyone
R2	Overloading deriving from other subjects	7	7	49	Plan and organize well and divide the work load evenly on all the weeks.	Project manager
R3	Lack of toolkit knowledge	9	8	72	Use time required to study up on the technology and understand it.	Everyone
R4	Internal conflicts	2	7	14	Reduce confrontation and negotiations by including a third party person like the supervisors	Project manager

R5	Team members being too late for meetings	5	3	15	Use punishment as a tool to make people be punctual.	Scrum master
R6	Language problems	2	5	10	Mainly use English as a common ground. Have patience with one another.	Everyone
R7	Customer changes the requirements	3	7	21	Have often contact with the customer so that we can have the same understanding about the requirements.	Customer contact
R8	Internal misunderstandings	5	5	25	Have meetings and talk with each other on the different parts of the system to make sure we have the same understanding of the project.	Technical manager

Appendix D

Test Collection

Test ID - Name	T1 - Basic GUI running
Description	GUI elements with a menu system shall be shown in a window using the Qt GUI framework. ITK and VTK libraries shall also be included.
Criteria	1. When the application is started, the user should be able to navigate in the menu system.
Estimated hours	69
Actual hours	79
Results	Passed
Comments	This was a very early prototype of the framework for the application, but the underlying design is in place which we can build on.

Test ID - Name	T2 - View 3D image
Description	The application shall be able to render the medical image parts in 3D.
Criteria	1. The user selects which medical images that will be 3D rendered. 2. The user shall then see the 3D rendering of the loaded medical image parts.
Estimated hours	84
Actual hours	80
Results	Passed
Comments	Managed to render and view a 3D image from the DICOM images, but needs optimization to get faster especially start rendering. Also the customer wants a surface rendering of only the skin, so it needs some improvements.

Test ID - Name	T3 - Read position and integrate the device
Description	The application should be able to read the position with the Phantom Omni device in our application.
Criteria	The user moves the Phantom Omni device and shall see the pointer move accordingly.
Estimated hours	73
Actual hours	103
Results	Passed
Comments	Position is read in the application, but will be opened in another window since QuickHaptics does not support integration with Qt at this moment.

Test ID - Name	T4 - Implement force feedback
Description	The Phantom Omni Probe shall receive force feedback if you hit an object in the application.
Criteria	The user moves the Phantom Omni device and tries to "enter" inside the object. The user shall then perceive vibration in the Phantom Omni device.
Estimated hours	68
Actual hours	8
Results	Passed
Comments	The force feedback works really well.

Test ID - Name	T5 - Optimize the 3D rendering
Description	The application shall be able to surface render medical images without including what is "inside" the object.
Criteria	1. The user selects which medical images that will be 3D rendered. 2. The user shall then see a 3D model of the loaded medical image parts.
Estimated hours	64
Actual hours	52
Results	Passed
Comments	Works really good, and is much faster than the rendering we had in sprint 1.

Test ID - Name	T6 - Generate image slices
Description	The application should be able to view and render image slices in realtime.
Criteria	The user navigates in the 3D model. Then, the application generates an image slice based on the position of the cursor in the 3D model.
Estimated hours	72
Actual hours	102
Results	Passed
Comments	This works in both haptic view and regular 3D view in Qt.

Test ID - Name	T7 - Load a case
Description	The user should be able to load an old case.
Criteria	The user chooses “File” in the menu and selects “Load case”. Then, the user shall be able to import an old case stored on the computer.
Estimated hours	30
Actual hours	24
Results	Passed
Comments	The user can not use the CT slice view if an OBJ file is loaded. This is because an OBJ file does not have anything inside since it is just a 3D surface.

Test ID - Name	T8 - Save a case
Description	The user should be able to save the 3D model it has created.
Criteria	The user chooses “File” in the menu and selects “Export surface”. Then, the user is able to select where the 3D model is going to be stored on the computer as an OBJ file.
Estimated hours	16
Actual hours	20
Results	Passed
Comments	None

Test ID - Name	T9 - All platforms compatible
Description	The application should run on Windows, Mac and Linux.
Criteria	The administrator uses CMake to build the project on the desired platform. Then the administrator builds and starts the program.
Estimated hours	94
Actual hours	94
Results	Passed
Comments	This is more a way to build the program on the major platforms.

Appendix E

Product Backlog

E.1 Actual Product Backlog

ID	Name	Importance	Estimate (hours)	Sprint	Description
P1	Basic GUI running	100	7	1	Be able to show basic GUI elements with a menu system as shown in the GUI prototype which the customer gave us
P2	View 3D image	90	84	1	Be able to render the medical image parts in 3D
P3	Read the position information	90	7	1	Get familiar with the Phantom Omni device and be able to read the position with the device
P4	Integrate the Phantom Omni device with the application	90	66	1	The user should be able to use the Phantom Omni device with our application
P5	Help access	20	3	3	The user can access help information from the help menu
P6	Load a case	50	30	3	The user should be able to load an old case
P7	All platforms compatible	50	94	3	The application should run on Windows, Mac and Linux

P8	View 3D image, CT slice and ultrasound image	70	196	3	The user should be able to view the 3D image, CT slice and ultrasound images in the same screen
P9	Implement user mode	10	24	3	Implement the different user modes like student/training, instructor/setup and test/evaluation mode
P10	Input parameter interface	30	24	3	Provide an interface for the following input parameters: depth, width, gain, and frequency
P11	Select from different curriculum	20	3	3	User should be able to select from different curriculum options: Trauma, Fast, Ultrasound, Anatomy etc.
P12	Save a case	70	16	3	Implement a function so the user can save the case that he/she is working with
P13	Render ultrasound images	80	24	3	The application should be able to render ultrasound images in real-time

P14	View CT slice images	80	72	2	The application should be able to view and render CT slice images in real-time
P15	Realistic ultrasound images	50	36	3	Make realistic ultrasound images from the CT slice image, where bones and different obstacles in the body is showed realistically on the screen
P16	Integrate Qt with ITK/VTK	90	62	1	Integrate the GUI with ITK and VTK
P17	Implement force feedback to the Phantom Omni Probe	50	68	2	The application should be able to give force feedback to the Phantom Omni Probe if it hits bones or other materials when the user navigates
P18	Optimize the 3D rendering	70	64	2	Make the 3D rendering go faster when the user is navigating the 3D object
P19	Extend the device support	60	64	2	Generalize the input device interface, so it works with most commercials haptic devices

E.2 Original Product Backlog

ID	Name	Importance	Estimate (hours)	Sprint	Description
P1	Basic GUI running	100	7	1	Be able to show basic GUI elements with a menu system as shown in the GUI prototype which the customer gave us
P2	View 3D image	90	84	1	Be able to render the medical image parts in 3D
P3	Read the position information	90	7	1	Get familiar with the Phantom Omni device and be able to read the position with the device
P4	Integrate the Phantom Omni device with the application	90	66	1	The user should be able to use the Phantom Omni device with our application
P5	Help access	10	5	4	The user can access help information from the help menu
P6	Load a case	50	24	2	The user should be able to load an old case
P7	All platforms compatible	50	12	4	The application should run on Windows, Mac and Linux

P8	View 3D image, CT slice and ultrasound image	70	24	3	The user should be able to view the 3D image, CT slice and ultrasound images in the same screen
P9	Implement user mode	30	24	2	Implement the different user modes like student/training, instructor/setup and test/evaluation mode
P10	Input parameter interface	30	24	3	Provide an interface for the following input parameters: depth, width, gain, and frequency
P11	Select from different curriculum	10	12	4	User should be able to select from different curriculum options: Trauma, Fast, Ultrasound, Anatomy etc
P12	Save a case	70	12	3	Implement a function so the user can save the case that he/she is working with
P13	Render ultrasound images	80	24	3	The application should be able to render ultrasound images in real-time

P14	View CT slice images	80	36	2	The application should be able to render CT slice images in real-time
P15	Realistic ultrasound images	50	36	4	Make realistic ultrasound images from the CT slice image, where bones and different obstacles in the body is showed realistically on the screen
P16	Integrate Qt with ITK/VTK	90	62	1	Integrate the GUI with ITK and VTK
P17	Implement force feedback to the Phantom Omni Probe	40	90	4	The application should be able to give force feedback to the Phantom Omni Probe if it hits bones or other materials when the user navigates

Appendix F

Sprint Backlog

F.1 Sprint 1

	Name	Component	Importance		Days in:													
				Date:	20/09	21/09	22/09	23/09	24/09	25/09	26/09	27/09	28/09	29/09	30/09	01/10	02/10	03/10
				Effort left:	288	278	274	247	242	210	205	205	158	149	154	58	30	30
Backlog item 1	Basic GUI running			7														
	Design menu bar	Basic GUI running	50		4	4	4	4	4	0	0	0	0	0	0	0	0	0
	Add customer logo	Basic GUI running	20		1	1	1	1	1	1	1	1	1	1	1	0	0	0
	Test	Basic GUI running	80		2	2	2	2	2	1	1	1	1	1	1	0	0	0
Backlog item 2	View 3D image			84														
	Render the medical images	View 3D image	100		52	44	44	32	32	20	20	20	10	10	0	0	0	0
	Choose the best volume mapper	View 3D image	90		10	10	10	10	10	6	6	6	4	4	2	0	0	0
	View 3D image with VTK	View 3D image	100		12	12	12	12	12	12	12	12	5	5	2	0	0	0
	View 3D image with Qt	View 3D image	100		2	2	2	2	2	2	2	2	2	2	0	0	0	0
	Test	View 3D image	70		8	8	8	8	8	8	8	8	6	6	2	0	0	0
Backlog item 3	Read the position information			7														
	Try the examples	Read the position	60		2	2	0	0	0	0	0	0	0	0	0	0	0	0
	Read the positions of sample model	Read the position	80		5	5	3	0	0	0	0	0	0	0	0	0	0	0
Backlog item 16	Integrate Qt with ITK/VTK			62														
	Integrate Qt with Microsoft Visual studio	Qt with ITK/VTK	100		36	36	36	30	30	30	30	30	15	15	0	0	0	0

	Name	Component	Importance		Days in:													
				Date:	20/09	21/09	22/09	23/09	24/09	25/09	26/09	27/09	28/09	29/09	30/09	01/10	02/10	03/10
				Effort left:	288	278	274	247	242	210	205	205	158	149	154	58	30	30
	Compile VTK	Qt with ITK/VTK	100		12	12	12	12	12	6	6	6	6	6	0	0	0	0
	Compile ITK	Qt with ITK/VTK	100		6	6	6	6	6	3	3	3	3	3	0	0	0	0
	Test	Qt with ITK/VTK	60		8	8	8	8	8	6	6	6	3	3	0	0	0	0
Backlog item 4	Integrate the Phantom Omni device with the application			66														
	Find suitable export format for 3D image	Integrate the device	100		6	6	6	6	6	6	6	6	4	0	0	0	0	0
	Export 3D image to .OBJ file	Integrate the device	80		40	40	40	40	40	40	40	40	40	40	20	10	10	10
	Read the positions of generated .OBJ file	Integrate the device	60		10	10	10	10	10	10	10	10	10	10	10	10	10	10
	Test	Integrate the device	40		10	10	10	10	10	10	10	10	10	10	10	10	10	10
Not in backlog	Report writing			42														
	Write Sprint 1 - Introduction	Report	60		2	0	0	0	0	0	0	0	0	0	0	0	0	0
	Design sprint backlog	Report	90		4	4	4	0	0	0	0	0	0	0	0	0	0	0
	Write Sprint 1 - Design chapters	Report	90		8	8	8	6	6	6	6	6	0	0	0	0	0	0
	Write Sprint 1 - Implementation chapter	Report	60		12	12	12	12	12	12	12	12	12	12	12	12	0	0
	Write Sprint 1 - Testing chapter	Report	50		8	8	8	8	8	8	8	8	8	8	8	8	0	0
	Write Sprint 1 - Result chapter	Report	30		4	4	4	4	4	4	4	4	4	4	4	4	0	0
	Write Sprint 1 - Evaluation chapter	Report	60		4	4	4	4	4	4	4	4	4	4	4	4	0	0
	Write Preliminary studies chapter	Report	30		20	20	20	20	15	15	10	10	10	5	5	0	0	0

F.2 Sprint 2

	Name	Component	Importance	04/10 Effort left:	Days in: 04/10	05/10	06/10	07/10	08/10	09/10	10/10	11/10	12/10
					462	436	400	364	344	338	338	320	292
Backlog item 4	Integrate the Phantom Omni device with the application			30									
	Export 3D image to .OBJ file	Integrate the device	90		10	0	0	0	0	0	0	0	0
	file	Integrate the device	90		10	10	0	0	0	0	0	0	0
	Test	Integrate the device	60		10	10	10	0	0	0	0	0	0
Backlog item 14	View CT slice images			72									
	Extract image slice	View CT slice images	100		18	18	18	18	18	18	18	18	18
	Integrate image slice in Qt	View CT slice images	90		24	24	24	24	24	24	24	24	24
	Improve extraction algorithm	View CT slice images	60		24	24	24	24	24	24	24	24	24
	Test	View CT slice images	60		6	6	6	6	6	6	6		
Backlog item 18	Optimize the 3D rendering			64									
	Try different compression algorithms	Optimize the 3D rendering	100		30	30	30	30	30	30	30	18	10
	Compare and choose the best one	Optimize the 3D rendering	70		10	10	10	10	10	10	10	10	10
	Create smaller .OBJ file	Optimize the 3D rendering	90		30	30	30	30	30	30	30	30	30
	Test	Optimize the 3D rendering	60		4	4	4	4	4	4	4	4	4
Backlog item 19	Extend the device support			64									
	Test Chai 3D with device	Extended the device support	80		15	15	15	15	15	15	15	15	15
	Test Open Haptics with device	Extended the device support	80		15	15	15	15	15	15	15	15	15
	Compare and choose the best one	Extended the device support	100		34	34	34	34	34	34	34	34	34
Backlog item 17	Implement force feedback to the Phantom Omni Probe			68									
	Include Quick Haptic standard for the force feedback in the application	Implement force feedback to the Phantom Omni Probe	100		68	68	68	68	68	68	68	68	68
Not in backlog	Report writing			164									
	Write Sprint 2 - Introduction	Report	60		2								
	Design sprint backlog	Report	90		4	4	4	4	4	4	4	4	4
	Write Sprint 2 - Design chapters	Report	90		8	8	8	8	8	8	8	8	8
	chapter	Report	60		12	12	12	12	12	12	12	12	12
	Write Sprint 2 - Testing chapter	Report	50		8	8	8	8	8	8	8	8	8
	Write Sprint 2 - Result chapter	Report	30		4	4	4	4	4	4	4	4	4
	Write Sprint 2 - Evaluation chapter	Report	60		4	4	4	4	4	4	4	4	4
	subchapters	Report	70		62	42	24	16	4	2	2	2	0
	Fix errors in report based on feedback	Report	90		38	34	26	10	6	4	4	4	0
	Finalize report for pre-delievery	Report	70		22	22	22	20	16	14	14	14	0

	Name	Component	13/10 270	14/10 256	15/10 244	16/10 244	17/10 242	18/10 170	19/10 148	20/10 134	21/10 118	22/10 84	23/10 55	24/10 30
Backlog item 4	Integrate the Phantom Omni device with the application													
	Export 3D image to .OBJ file	Integrate the device	0	0	0	0	0	0	0	0	0	0	0	0
	file	Integrate the device	0	0	0	0	0	0	0	0	0	0	0	0
	Test	Integrate the device	0	0	0	0	0	0	0	0	0	0	0	0
Backlog item 14	View CT slice images													
	Extract image slice	View CT slice images	18	18	18	18	18	18	18	12	10	6	0	0
	Integrate image slice in Qt	View CT slice images	24	24	24	24	24	24	24	24	24	20	16	12
	Improve extraction algorithm	View CT slice images	24	24	24	24	24	24	24	24	24	24	24	18
	Test	View CT slice images												
Backlog item 18	Optimize the 3D rendering													
	Try different compression algorithms	Optimize the 3D rendering	0	0	0	0	0	0	0	0	0	0	0	0
	Compare and choose the best one	Optimize the 3D rendering	7	0	0	0	0	0	0	0	0	0	0	0
	Create smaller .OBJ file	Optimize the 3D rendering	30	28	16	16	16	16	0	0	0	0	0	0
	Test	Optimize the 3D rendering	3	2	2	2	2	2	0	0	0	0	0	0
Backlog item 19	Extend the device support													
	Test Chai 3D with device	Extended the device support	15	15	15	15	14	12	10	6	0	0	0	0
	Test Open Haptics with device	Extended the device support	15	15	15	15	14	12	10	6	0	0	0	0
	Compare and choose the best one	Extended the device support	34	34	34	34	34	34	34	34	32	20	6	0
Backlog item 17	Implement force feedback to the Phantom Omni Probe													
	Include Quick Haptic standard for the force feedback in the application	Implement force feedback to the Phantom Omni Probe	68	68	68	68	68	0	0	0	0	0	0	0
Not in backlog	Report writing													
	Write Sprint 2 - Introduction	Report												
	Design sprint backlog	Report	0	0	0	0	0	0	0	0	0	0	0	0
	Write Sprint 2 - Design chapters	Report	4	0	0	0	0	0	0	0	0	0	0	0
	chapter	Report	12	12	12	12	12	12	12	12	12	6	4	0
	Write Sprint 2 - Testing chapter	Report	8	8	8	8	8	8	8	8	8	4	3	0
	Write Sprint 2 - Result chapter	Report	4	4	4	4	4	4	4	4	4	2	1	0
	Write Sprint 2 - Evaluation chapter	Report	4	4	4	4	4	4	4	4	4	2	1	0
	subchapters	Report	0	0	0	0	0	0	0	0	0	0	0	0
	Fix errors in report based on feedback	Report	0	0	0	0	0	0	0	0	0	0	0	0
	Finalize report for pre-delievery	Report	0	0	0	0	0	0	0	0	0	0	0	0

F.3 Sprint 3

	Name	Component	Importance	in:									
				Date:	25/10	26/10	27/10	28/10	29/10	30/10	31/10	01/11	02/11
				Effort left:	462	427	388	346	309	299	299	291	242
Backlog item 14	View CT slice images			30									
	Improve extraction algorithm	View CT slice images	60		12	2	0	0	0	0	0	0	0
	Test	View CT slice images	60		18	16	8	0	0	0	0	0	0
Backlog item 5	Help access			3									
	future expansion	Help access	30		2	2	2	2	2	2	2	2	2
	Test	Help access	20		1	1	1	1	1	1	1	1	1
Backlog item 6	Load a case			30									
	GUI interface	Load a case	50		5	5	5	5	5	5	5	5	5
	Connect Qt with file or folder	Load a case	70		10	10	10	10	10	10	10	10	10
	Call the appropriate reader	Load a case	70		10	10	10	10	10	10	10	10	10
	Test	Load a case	50		5	5	5	5	5	5	5	5	5
Backlog item 7	All platforms compatible			94									
	Create CMake file	All platforms compatible	70		84	84	84	84	84	84	84	84	84
	Test	All platforms compatible	50		10	10	10	10	10	10	10	10	10
Backlog item 8	View 3D image and CT slice			196									
	Integrate image slice (implemented in VTK) with Qt	View 3D image and CT slice	100		80	60	50	32	16	16	16	8	0
	Integrate image slice (implemented in VTK) with Open Haptics	View 3D image and CT slice	100		90	90	85	79	70	70	70	58	50
	Test	View 3D image and CT slice	70		26	26	26	26	24	24	24	22	16
Backlog item 11	Select from different curriculum			3									
	Establish an infrastructure for future expansion	Select from different curriculum	30		2	2	2	2	2	2	2	2	2
	Test	Select from different curriculum	20		1	1	1	1	1	1	1	1	1
Backlog item 12	Save a case			16									
	Save volum as OBJ file		70		12	12	12	12	12	12	12	12	6
	Test		60		4	4	4	4	4	4	4	4	4
Not in backlog	Report writing			90									
	Write Sprint 3 - Introduction	Report	60		2	2	0	0	0	0	0	0	0
	Design sprint backlog	Report	90		4	4	2	0	0	0	0	0	0
	Write Sprint 3 - Design chapter	Report	90		8	8	8	8	8	8	8	8	8
	chapter	Report	60		12	12	12	12	12	12	12	12	12
	Write Sprint 3 - Testing chapter	Report	50		8	8	8	8	8	8	8	8	8
	Write Sprint 3 - Result chapter	Report	30		4	4	4	4	4	4	4	4	4
	Write Sprint 3 - Evaluation chapter	Report	60		4	4	4	4	4	4	4	4	4
	chapter	Report	90		33	30	20	15	10	0	0	0	0
	Write user documentation	Report	10		15	15	15	12	7	7	7	7	0

	Name	Component	03/11	04/11	05/11	06/11	07/11	08/11	09/11	10/11	11/11	12/11	13/11	14/11
			225	192	160	160	160	126	75	39	15	0	0	0
Backlog item 14	View CT slice images													
	Improve extraction algorithm	View CT slice images	0	0	0	0	0	0	0	0	0	0	0	0
	Test	View CT slice images	0	0	0	0	0	0	0	0	0	0	0	0
Backlog item 5	Help access													
	future expansion	Help access	2	2	2	2	2		2	2	0	0	0	0
	Test	Help access	1	1	1	1	1	1	1	1	0	0	0	0
Backlog item 6	Load a case													
	GUI interface	Load a case	5	5	5	5	5	0	0	0	0	0	0	0
	Connect Qt with file or folder	Load a case	10	10	10	10	10	9	0	0	0	0	0	0
	Call the appropriate reader	Load a case	10	10	10	10	10	10	10	0	0	0	0	0
	Test	Load a case	5	5	5	5	5	5	5	5	0	0	0	0
Backlog item 7	All platforms compatible													
	Create CMake file	All platforms compatible	84	74	60	60	60	46	32	16	8	0	0	0
	Test	All platforms compatible	10	10	10	10	10	10	10	9	5	0	0	0
Backlog item 8	View 3D image and CT slice													
	Integrate image slice (implemented in VTK) with Qt	View 3D image and CT slice	0	0	0	0	0	0	0	0	0	0	0	0
	Integrate image slice (implemented in VTK) with Open Haptics	View 3D image and CT slice	38	23	10	10	10	5	0	0	0	0	0	0
	Test	View 3D image and CT slice	15	13	10	10	10	10	0	0	0	0	0	0
Backlog item 11	Select from different curriculum													
	Establish an infrastructure for future expansion	Select from different curriculum	2	2	2	2	2	2	0	0	0	0	0	0
	Test	Select from different curriculum	1	1	1	1	1	1	0	0	0	0	0	0
Backlog item 12	Save a case													
	Save volum as OBJ file		2	0	0	0	0	0	0	0	0	0	0	0
	Test		4	0	0	0	0	0	0	0	0	0	0	0
Not in backlog	Report writing													
	Write Sprint 3 - Introduction	Report	0	0	0	0	0	0	0	0	0	0	0	0
	Design sprint backlog	Report	0	0	0	0	0	0	0	0	0	0	0	0
	Write Sprint 3 - Design chapter	Report	8	8	6	6	6	0	0	0	0	0	0	0
	chapter	Report	12	12	12	12	12	9	4	0	0	0	0	0
	Write Sprint 3 - Testing chapter	Report	8	8	8	8	8	8	3	0	0	0	0	0
	Write Sprint 3 - Result chapter	Report	4	4	4	4	4	4	4	2	0	0	0	0
	Write Sprint 3 - Evaluation chapter	Report	4	4	4	4	4	4	4	4	2	0	0	0
	chapter	Report	0	0	0	0	0	0	0	0	0	0	0	0
	Write user documentation	Report	0	0	0	0	0	0	0	0	0	0	0	0

Appendix G

Timesheet

Phases	Week	34	35	36	37	38	39	40
Project Planning	Estimated / Spent	144 / 140	144 / 148	144 / 150	144 / 102			
Sprint 1						144 / 136	144 / 170	
- Preliminary studies	Estimated / Spent					62 / 30	10 / 4	
- Requirements	Estimated / Spent					62 / 30	0 / 0	
- Documentation	Estimated / Spent					10 / 10	50 / 40	
- Development	Estimated / Spent					10 / 64	84 / 126	
Sprint 2								144 / 147
- Preliminary studies	Estimated / Spent							16 / 15
- Requirements	Estimated / Spent							14 / 10
- Documentation	Estimated / Spent							62 / 118
- Development	Estimated / Spent							52 / 4
- Testing	Estimated / Spent							0 / 0
Sprint 3								
- Preliminary studies	Estimated / Spent							
- Requirements	Estimated / Spent							
- Documentation	Estimated / Spent							
- Development	Estimated / Spent							
- Testing	Estimated / Spent							
presentation								
Total	Estimated / Spent	144 / 140	144 / 148	144 / 150	144 / 102	144 / 136	144 / 170	144 / 147

Phases	Week	41	42	43	44	45	46	47
Project Planning	Estimated / Spent							
Sprint 1								
- Preliminary studies	Estimated / Spent							
- Requirements	Estimated / Spent							
- Documentation	Estimated / Spent							
- Development	Estimated / Spent							
Sprint 2		144 / 145	144 / 132					
- Preliminary studies	Estimated / Spent	20 / 30	20 / 20					
- Requirements	Estimated / Spent	27 / 30	0 / 0					
- Documentation	Estimated / Spent	62 / 57	10 / 8					
- Development	Estimated / Spent	28 / 20	106 / 98					
- Testing	Estimated / Spent	8 / 8	8 / 6					
Sprint 3				144 / 154	144 / 132	144 / 152		
- Preliminary studies	Estimated / Spent			0 / 0	0 / 0	0 / 0		
- Requirements	Estimated / Spent			10 / 10	0 / 0	0 / 0		
- Documentation	Estimated / Spent			50 / 60	50 / 50	60 / 68		
- Development	Estimated / Spent			74 / 74	74 / 62	74 / 74		
- Testing	Estimated / Spent			10 / 10	20 / 20	10 / 10		
presentation							144 / 140	144 / 156
Total	Estimated / Spent	144 / 145	144 / 132	144 / 154	144 / 132	144 / 152	144 / 140	144 / 156